工學碩士 學位論文

WIPI 플랫폼상에서 문자 대화 서비스 및 상용어구 기능의 설계 및 구현

Design and Implementation of Chatting Service and Common Phrase Function on WIPI Platform

指導敎授 林 宰 弘

2005年 2月

韓國海洋大學校 大學院

電子通信工學科

林 昶 默

목 차

그 림 목 차

표 목 차

Abstract

Recently, domestic interest about wireless internet is rising gradually. Technologies about new mobile communication equipment or wireless contents are presented and displayed constantly through news or other various medias.

Characteristic of present domestic wireless market is that mobile communication businessmans appropriate each other different mobile platform.

So, contents provider's development environment can not be same each other. This makes contents providers repeat same work. Also, it is hard that user freely uses contents that the different mobile communication company offers. To solve this problem, standardization work of wireless internet market was begun. The three mobile communication companies and TTA (Telecommunications Technology Association), RRU(Radio Research Laboratory), ETRI(Electronics and Telecommunications Research Institute) progressed standardization. By the result, May 2002, WIPI(Wireless Internet Platform for Interoperability) that is wireless standard platform selected by mobile platform standard that is TTA organization standard.

In this paper, I am going to examine the WIPI. WIPI can reduce mostly expense that happen when we use different platform. And, in this paper, I am going to discuss convenience in chatting service between cellular phones. For this, I design and embody common use phrase function. Through this, I am

going to show improvement of the character input speed in cellular phone. And I wish to discuss expected cost decrease effect. Investigate about platform of treatise that is used in domestic. analyze about the characteristic, merits and demerits. Chatting system and common phrase function design and embody. Finally, wish to discuss about advantage of common use phrase function and practical use field.

제 1 장 서 론

최근 급속한 무선통신의 발달은 무선 및 이동통신 기술에 유선 데이터 통신의 전유물이었던 인터넷의 확장을 가져왔으며, 이러한 현상은 인터넷을 통해 유·무선 통합개념을 도출하였고 기존 인터넷 기술은 유선 인터넷의 한계인 이동성 및 편이성을 확보할 수 있게 되었다.

물론 아직까지는 유·무선 통합의 전이단계에 불과하지만 가까운 미래에 유·무선의 완벽한 인터넷 통합을 통해 유·무선 구별 없는 통합적이고 보편적인 인터넷 시대가 마련될 것으로 예상된다. 이러한 예측은 최근 CDMA(Code Division Multiple Access), PCS(Personal Communications Services), 무선 LAN(Local Area Network) 등 전파통신 기술의 발달과 함께 무선 통신기술의 제한 구조 극복 및 유·무선 통신기술과의 연동, 유선과 무선과의 연계를 위한 무선 소프트웨어적인 인터넷 기술요소 개발 및 첨가로 인해 점차 통합된 유·무선 포털 서비스가 가능할 것으로 기대되고 있다. 특히 전파 통신기술에 있어 광대역화 및 디지털화 기술의 발달과 유·무선망 연동 및 망 융합화의 경우 인터넷 확산을 가속화 할 수 있는 주요 요소로 작용하고 있다[1],[2].

뿐만 아니라 이러한 추세와 더불어 컴퓨팅 파워의 급속한 향상과 관련, 주요 부품의 소형화와 단순화로 인한 전체적인 기기 자체의 소형화가 이루어지고 있고 그에 따른 부가적인 소프트웨어 기술개발도 활발하고 다양하게 성장하고 있다. 이러한 효과로 인해 현재 기술 수준은 모바일 시스템의 경우 개인용 컴퓨터 개발 역사 중 초기시스템 보다 더 향상된 기능을 가지고 있으며 최근에는 컬러 기술을 바탕으로 그 이상의 기능도 가능할 것으로 예측하고 있다.

더불어 무선 인터넷의 광역화된 개념은 IMT-2000(International Mobile Telecommunications - 2000) 시스템과 연계된 이동 무선 인터넷과 유선의 연장개념으로서의 무선 LAN을 기반으로 한 고정 무선 인터넷을 통틀어 이야기하기도 한다.

이와 같은 외형적인 기능 발달과 함께 내재적인 시스템 콘텐츠 발달의 경우 이동 무선 인터넷 중 서비스 극대화를 이루기 위한 단말기내의 중간자적 역할인 모바일 플랫폼이라는 새로운 개념의 미들웨어를 주목할 필요가 있다[3].

무선 인터넷의 기능 중 운영적인 측면에서 핵심적인 역할을 담당하게 될 모바일 플랫폼은 향후 지속적인 유무선 인터넷의 통합을 위한 중요한 역할을 담당할 것이다.

그러나 무선 인터넷 망에 따라 발생되는 제한적이고 획일적인 콘텐츠 서비스와 서로 다른 플랫폼의 사용이 문제점으로 대두되었다.

이러한 획일적인 콘텐츠 서비스와 콘텐츠 개발자들의 문제를 해결하기 위해 WIPI(Wireless Internet Platform for Interoperability)라는 표준 모바일 플랫폼이 개발되게 되었다[4],[5].

본 논문에서는 이러한 WIPI 환경에서 휴대폰간의 문자 전송 서비스와 이에 따른 상용어구 기능의 적용 기법을 통한 입력 시스템 체계의 효율성을 고려하고, 장애인이나 노령 사용자층의 휴대폰 사용시 키 입력의 불편함을 해소하고 친숙해질 수 있도록 상용어구 기능을 설계 및 구현하였다.

본문의 구성은 제 2 장에서는 현재 모바일 플랫폼의 문제와 WIPI의 개발배경 그리고 WIPI의 구조, 특징, 주요규격에 대해 알아보고, 제 3 장과 제 4 장에서는 WIPI 플랫폼상에서 문자 대화 서비스와 사용자에게 문자 대화시에 편리함을 제공할 수 있는 상용어구 기능을 설계 및 구현하였고, 제 5 장에서는 결론을 기술하였다.

제 2 장 모바일 플랫폼 개요

2.1 모바일 플랫폼

모바일 플랫폼이란 모바일 기기에 내장되는 시스템으로 하드웨어와 어플리케이션 중간에 위치하여 어플리케이션이 실행될 수 있는 환경을 말한다.

우리나라에서 현재 사용되고 있는 무선 인터넷 플랫폼의 종류는 SKT의 GVM(Game Virtual Machine), SK-VM(SK-Virtual Machine), LGT의 KVM(Kilobyte Virtual Machine), KTF의 MAP(Mobile Application S/W Plus-In), BREW(Binary Runtime Environment for Wireless) 등이 있다.

<표 2-1> 국내 모바일 플랫폼의 현황

플랫폼	서비스 사업자	개발환경	개발사	수행방법
GVM	SKT	C/C++	신지소프트	스크립트
SK_VM	SKT	자바	XEC	스크립트
MAP	KTF	C/C++	모빌탑	바이너리
BREW	KTF	C/C++	퀄컴	바이너리
KVM	LGT	자바	아로마소프트	스크립트

<표 2-1>은 개발환경과 수행방법에 따라 모바일 플랫폼을 분류한 것이다. 무선 인터넷 플랫폼에 사용되는 언어는 단말기의 특성을 고려하여 기술적으로 이진코드(Binary Code)를 실행시키는 C 언어 계열과 버추얼 머신에서 인터프리트(Interpret) 하는 과정을 거치는 자

바 언어 계열로 나뉘어져 있다.

버추얼 머신은 특정 응용 프로그램을 실행시키기 위한 하나의 실행 과정이라고 말할 수 있다. 버추얼 머신의 가장 큰 특성은 이식성과 API(Application Programming Interface)의 추상화를 할 수 있으므로 사양이 다른 핸드폰이라도 같은 API를 사용하여 프로그래머는 프로그램의 개발을 자유롭게 할 수 있다. 또한 하위레벨의 환경에 신경 쓸 필요가 없다.

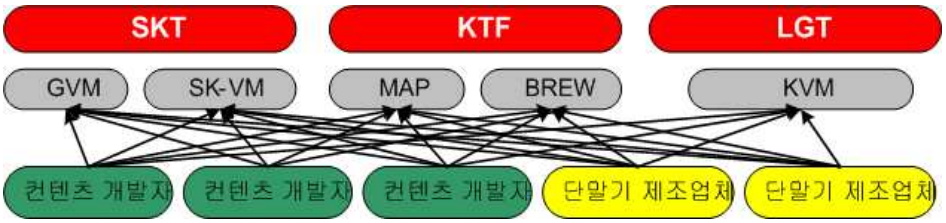
미들웨어의 플랫폼 방식은 크게 바이너리 다운로드(Binary Download)방식과 스크립트(Script)방식이 있는데, 다운로드방식은 어플리케이션을 직접 읽고 직접 실행하는 것이 아닌 어플리케이션 자체가 구동할 수 있도록 지원하는 방식이며, 스크립트 방식은 상위 어플리케이션 코드를 직접 읽어서 명령을 실행하는 방식이다. 바이너리 다운로드 방식의 대표적인 예는 MAP, BREW이며 스크립트 방식에는 SK-VM, KVM, GVM 등이 있다[6].

2.2 모바일 표준 플랫폼 WIPI

2.2.1 WIPI의 등장 배경

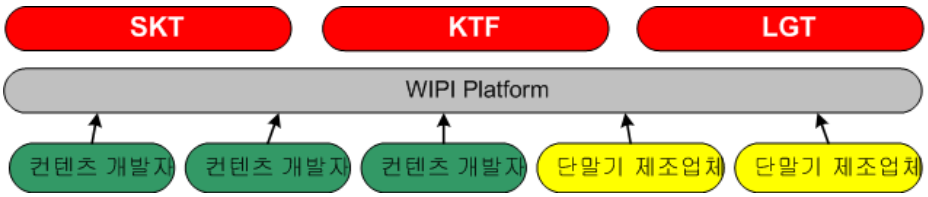
모바일 인터넷 시장에는 여러 방식의 플랫폼들이 난립해 있는 실정이다. J2ME(자바 언어), GVM(모바일 C), BREW(C/C++) 등의 여러 가지 플랫폼들이 자신만의 장점을 가지고 각각의 이동 통신사에서 상용화되어 있다.

하지만 실제 어플리케이션 개발자의 입장에서 보면 한 가지 아이템의 어플리케이션을 서비스하기 위해서 각 이동 통신사의 플랫폼에 맞추어서 여러 벌의 어플리케이션을 개발하여야 하기 때문에 개발 비용 및 기술자 수급문제에서 큰 문제점을 가지고 있다.



<그림 2-1> 플랫폼이 서로 다른 개발환경

무선 인터넷이 발전하기 위해서는 여러 종류의 다양한 어플리케이션들이 개발되어 사용자들의 선택을 풍부하게 만들어야 함에도 불구하고 앞에서 말한 여러 플랫폼의 서로 다른 제약 조건 때문에 발전이 훨씬 더디게 된다. 특히 소규모 콘텐츠 업체의 경우에는 훨씬 그 사정이 심각해지게 된다. <그림 2-1>은 현재 다양한 플랫폼 사용으로 인한 문제를 보여주고, <그림 2-2>는 통일된 WIPI 플랫폼을 사용할 경우의 개발환경의 향상을 보여준다.

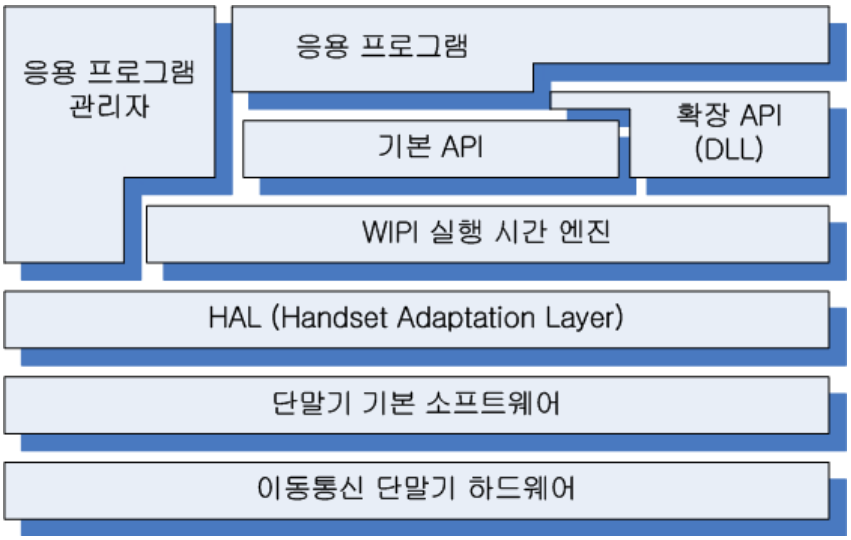


<그림 2-2> WIPI 플랫폼으로 통일된 개발환경

유선 인터넷 경우에는 HTML(HyperText Markup Language) 방식으로 통합되어 거의 모든 사람들이 동일한 방식으로 접속하고 있고 개발자 역시 동일한 형식으로 콘텐츠를 구성하는 것처럼 무선 인터넷에 있어서도 동일한 브라우저, 동일한 플랫폼을 가져야 한다는 것은 당연한 사실이다[7].

특히, 전 세계에서 플랫폼의 표준화가 이미 진행되었고 자바의 경우 버추얼 머신은 호환성이 전혀 없어 표준화의 빌미를 제공하였으며, 썬 마이크로시스템즈에서 개발한 소프트웨어 사용에 따른 로열티를 지불하고 있는 실정으로 각종 언어로 작성된 콘텐츠를 모두 지원하면서 퀄컴의 BREW를 능가할 정도로 경쟁력을 갖춘 표준 플랫폼의 개발을 기본 방향으로 정하였으며, 또한 버추얼 머신은 표준화 대상에서 제외하며 서버 측면과 단말기 측면을 모두 고려한 표준화를 추진하여 언어 변환 등 다양한 기능을 서버에서 처리하는 형태로 표준화가 진행되고 있다. 각 이동 통신사들의 요구사항을 종합해 본 결과 WIPI는 단일 플랫폼으로서 가져야 할 여러 가지 우수한 기능을 기본적으로 탑재하게 되었다.

2.2.2 WIPI의 구조



<그림 2-3> WIPI 플랫폼 구조

WIPI 규격에서 정의하는 모바일 표준 플랫폼의 구조는 <그림

2-3>과 같다. 그림 하단에 있는 단말기 기본 소프트웨어란 간단한 단말기 운영체제와 통신 기본 기능 및 각종 디바이스 드라이버가 포함된다. 단말기 기본 소프트웨어는 제조사에 따라 기능이나 규모가 다양할 수 있지만, 여기에 HAL(Handset Adaptation Layer) 계층을 두어 플랫폼이 바라보는 단말기 소프트웨어를 추상화할 수 있도록 하였다. WIPI에서 HAL의 표준화는 3GPP(Third Generation Partnership Project)에서도 획기적인 시도로 받아들이고 있다. 향후 이에 대한 논의가 심도 있게 논의 될 것으로 보인다. 응용 프로그램 개발자 입장에서 바라본 플랫폼은 기본 API 계층이다[8].

(1) HAL

플랫폼 이식에 있어서 하드웨어 독립성을 지원하기 위한 계층이다. 이를 통해 단말기에 대한 추상화가 이루어지고, 하드웨어가 독립적으로 플랫폼을 구성한다. 예를 들면, 국내 CDMA 단말기의 경우 퀄컴 운영체제(REX) 위에 HAL만을 포팅하면 단말기용 플랫폼이 되고, 윈도우는 HAL만을 포팅하면 윈도우용 에뮬레이터가 된다. 따라서 단말기 제조사가 한 번만 HAL API에 따라 구현해 두면, 다양한 플랫폼 구현이 빠른 시간 내에 포팅이 되는 장점이 있다.

(2) 기본 API

응용 프로그램 개발자가 사용하는 플랫폼에서 지원하는 기본 API 모음이다. C 및 자바 API로 구성되어 있고 C 및 자바 API는 기능면에서 동등한 API를 제공한다. 플랫폼 규격에서는 자바 언어용 응용 프로그램도 C 언어 응용 프로그램과 마찬가지로 바이너리로 수행 하도록 정하기 때문에 개발자는 선호하는 언어로 개발할 수 있다. 일반적으로 자바 언어로 대부분의 응용 프로그램을 제작할 수 있고, C 언어로는 속도에 아주 민감한 각종 멀티미디어 코덱이나 보

안 모듈 등을 제작할 것으로 기대된다.

(3) 단말기 기본 소프트웨어

간단한 단말기 운영체제와 통신 기본기능 및 각종 디바이스 드라이버들이 포함되어 있다.

(4) 응용 프로그램 관리자

응용 프로그램 다운로드, 설치, 삭제 등의 응용 프로그램을 관리하며, API 및 컴포넌트를 추가/갱신하는 역할을 수행한다.

(5) 동적 컴포넌트

어플리케이션 관리자를 통하여 추가/갱신된 API 및 컴포넌트를 지칭한다.

2.2.3 하드웨어 요구사항

규격에서 항상 언급되는 것이 단말기 사양이다. 즉, 어느 정도 수준의 단말기가 되어야 WIPI 플랫폼이 올라갈 수 있는가는 플랫폼 얘기가 나올 때 첫 번째로 관심을 갖는 궁금증이지만, 규격에서는 최소 권장 사양을 주로 언급하므로 지금 출시되고 있는 단말기 사양보다는 많이 낮을 수밖에 없다. 규격에 명시된 단말기 최소 권장 사양은 아래와 같다. 특정 CPU에 종속된 환경이 아니라 다양한 단말기를 모두 수용하기 위해서 아래의 최소한의 사양을 논의하여 정하였다.

(1) 디스플레이

가. 스크린 크기 : 95 × 54 이상

나. 색상 : 회색조 4가지 이상 또는 천연색 256가지 이상

(2) 입출력 장치

가. 입력 장치 : 키패드

나. 사운드 장치 : 진동 및 비프음

다. 네트워크 : 무선 및 시리얼을 통한 전송

(3) 비휘발성 메모리

가. 플랫폼 라이브러리가 사용할 수 있는 비휘발성 메모리 600KB 이상

나. 응용 프로그램 관리자 및 기본 응용 프로그램에서 사용할 수 있는 비휘발성 메모리 400KB 이상

다. 응용 프로그램이 사용 가능한 파일 시스템 공간으로 500KB 이상

(4) 휘발성 메모리

가. 응용 프로그램에서 사용 가능한 힙(HEAP) 영역으로 300KB 이상

나. 플랫폼 라이브러리에서 사용 가능한 영역으로 20KB 이상

2.2.4 주요 기능 및 규격

다음은 플랫폼이 갖추어야 할 주요 기능 규격에 대해 설명하고자 한다. 이 기능들은 기본 API를 통해 지원될 수 있는 부분과 플랫폼 내부에서 처리해야 하는 부분이 있게 된다. 이러한 기능들은 기존의 플랫폼 기능을 모두 수용하고, 기존의 플랫폼에서 적용되지 않았던 새로운 기능을 적극 수용하여 차세대 단말기에서 손색이 없도록 고려하였다. 다음과 같은 주요 특징을 가진다.

(1) WIPI의 주요기능

가. C 및 자바 언어로 작성된 응용 프로그램의 실행 환경을 제공한다.

나. 자바 언어로 작성된 프로그램에 대해 고속의 바이너리 코드 실행 환경을 제공한다.

다. 다중 응용 프로그램의 동시 실행 환경을 제공하며, 다중 응용 프로그램간 통신 기능을 제공한다.

라. 다운로드에 의한 동적 공유 라이브러리를 지원함에 따라, 동적으로 API를 추가/갱신하는 기능을 제공한다.

마. 고효율적인 메모리 관리

바. 메모리 압축 기능과 함께 메모리 자동 정리기능을 제공한다.

사. 응용 프로그램 종료시 자동으로 메모리 해제 기능을 제공한다.

(2) 응용 프로그램 머신 코드 규격

WIPI는 기존의 다양한 플랫폼의 기능을 수용하기 위하여, C 언어와 자바 언어를 위한 규격을 모두 지원한다. C 언어로 작성된 바이너리 형태의 응용 프로그램은 성능이 우수하다. 이러한 장점을 자바 언어에도 동일하게 적용하기 위하여, 자바 언어의 중간코드인 바이트 코드를 게이트웨이를 통해 안전한 서버에서 머신 코드 형태로 바꾼 뒤 다운로드되는 것을 가정하고, 머신 코드로 수행해야 됨을 규격에서 정하고 있다.

(3) 다중 응용 프로그램 수행

플랫폼은 동시에 여러 개의 응용 프로그램이 메모리에 적재되어 수행될 수 있는 환경을 지원하고 여러 개의 응용 프로그램을 동시에 실행할 수 있도록 하였다. 또한, 여러 개의 응용 프로그램 간 실

행 우선순위를 두고 매 순간 실행 가능한 가장 높은 우선순위의 응용 프로그램을 실행하도록 하였다.

여러 응용 프로그램을 수행할 경우 각 응용 프로그램은 독립적으로 수행되어야 하며, 독립적인 응용 프로그램간의 통신을 지원하기 위하여 공유 메모리와 이벤트를 전달할 수 있는 방법을 제공하도록 하였다.

이 기능이 구현되면 단말기에서 주소록을 보다가 주소록 내에서 이메일을 보내고자할 때 바로 메일 전송 응용 프로그램을 기동하고 주소록 내에 저장된 이메일 주소를 전달받아 메일을 보낼 수 있으며, 메일을 보낸 후 종료하면 다시 주소록 상태로 돌아올 수 있는 기능이 가능하게 된다. 또한, 이 기능을 응용하게 되면 마치 PC상의 Alt+⇆키에 의한 응용 프로그램 전환이 일어나듯이 동일한 기능이 단말기에서도 실행될 수 있게 된다.

(4) 지원 프로그래밍 언어

기존의 C 언어 응용 프로그램과 자바 언어 응용 프로그램을 모두 수용하기 위해서 복수 프로그래밍 언어를 지원한다. 플랫폼의 C 언어 지원을 보면, 기본 API에 정의된 C 언어용 API와 C 언어 문맥을 지원하도록 하였다. 단말기의 제한된 환경을 고려하여 플랫폼은 방대한 표준 C 라이브러리 중에서 주요 함수를 필수적으로 지원해야 함을 명시하고 있다.

자바 언어 지원의 경우는 기본 API에 정의된 자바 언어용 API와 자바 언어 문맥을 지원하도록 하였다. 자바 언어 응용 프로그램이 머신 코드로 수행되지만, 자바 언어가 가지고 있는 모든 문맥을 지원한다.

(5) 플랫폼 보안

단말기에는 보호되어야 할 개인 정보 데이터가 있으며, 항상 통화 대기 상태를 유지해야 하므로 응용 프로그램이 사용하는 플랫폼의 API와 단말기 내의 데이터에 대해서 접근 권한을 명확히 지정하여 관리해야 한다. 플랫폼은 다음 세 단계 보안수준을 정의한다.

가. 일반 수준

가장 낮은 수준의 보안 레벨로, 보통 신뢰할 수 없는 일반 개발자가 제공하는 응용 프로그램에 적용된다. 따라서 이 수준에서는 단말기에 영향을 미치거나, 개인 정보 등에 접근을 막도록 한다.

나. 콘텐츠 개발자 수준

이미 알려진 CP(Contents Provider)들은 어느 정도 신뢰할 수 있다고 보고, 단말기에 심각한 영향을 미치지 않는 범위 내에서 접근을 허용하도록 한다.

다. 시스템 수준

완전히 신뢰할 수 있는 것으로 보고, 모든 접근을 허용하도록 한다.

(6) API 보안

API별 보안 지원을 위해서 특정 API 그룹을 보안 대상 그룹으로 구분하여 해당 그룹별로 보안 수준을 지정하도록 되어 있다. 플랫폼은 이때 각 그룹에 대해서 다음의 접근 수준을 지정할 수 있다.

가. NO ACCESS : 허용하지 않음.

나. READ ONLY : 읽기만 허용함.

다. WRITE ONLY : 쓰기만 허용함.

라. READ/WRITE : 읽기, 쓰기 모두 허용함.

API 그룹의 접근 수준과 보안 수준 설정에 대한 정책은 이동통신사에 의해 플랫폼 이식 시점에 결정되어 적용되게 된다.

(7) 디렉터리 보안

플랫폼은 다음의 세 가지 디렉터리 접근 방식을 지원하도록 되어 있다.

가. 개인 디렉터리

응용 프로그램 관리자를 제외하고는 해당 응용 프로그램 자신만이 접근할 수 있는 디렉터리이다.

나. 응용 프로그램 공유 디렉터리

이미 서로 합의된 응용 프로그램들 간에 공유하기 위한 디렉터리이다.

다. 시스템 공유 디렉터리

응용 프로그램에 관계없이 공유되는 디렉터리이다.

(8) 서비스 보안

서비스 보안을 위해 자바 언어로 개발된 응용 프로그램들의 경우는 코드레벨의 안정성 및 보안성을 확인할 수 있는 방법을 적용한다. 또한, 신뢰할 수 있는 모든 응용 프로그램은 응용 프로그램 명세 파일과 실행 코드에 서명을 받도록 하는 방법을 적용한다. 이때, 서명은 유능하고 신뢰할 수 있는 기관이 맡게 되는데, 한국의 경우는 TTA(Telecommunications Technology Association)에서 그 역할을 할 것으로 기대된다.

(9) API 추가/갱신 지원 - 선택 규격

플랫폼은 API를 무선망을 통해서 추가/갱신할 수 있다. 추가/갱신된 API는 지속적으로 유지되어야 하며, 이를 지원하기 위하여 플랫폼은 API 추가/갱신에 따른 버전 관리 및 설치/삭제 기능을 갖도록 되어 있다. 플랫폼이 무선망을 통한 API 추가/갱신이 가능하도록 구

현할 경우 규격의 부속서에 명시한 API 규격에 따라 지원해야 하며, 추가/갱신된 API에 대해서도 API 보안수준 정책을 동일하게 적용하도록 하였다.

API 추가/갱신 지원 기능을 통해 다음의 서비스 시나리오가 가능할 것으로 기대 된다.

가. 동적으로 API의 추가/갱신을 지원함으로써 차별화된 API의 확장이 가능

나. 멀티미디어 코덱이나 보안 모듈 등을 동적 라이브러리 형태로 개발 가능

다. 사용자의 취향에 따른 모든 UI(User Interface) 컴포넌트를 동적으로 변경 가능

라. 플랫폼이 출시된 후 버그 패치 혹은 기능 추가가 가능

(10) 메모리 관리

플랫폼은 제한된 메모리를 가지고 있고, 다중 응용 프로그램이 수행되고 있으므로 효율적인 메모리 관리 기능이 요구된다. 응용 프로그램이 사용하는 메모리를 다음과 같이 관리하도록 하였다.

가. 자동 메모리 해제

하나의 응용 프로그램이 종료되면, 해당 프로그램과 관련된 모든 메모리는 플랫폼에 반환해야 한다. 따라서 이벤트 등 각종 동적으로 사용된 메모리는 자동으로 모두 반환해야만 하도록 되어 있다.

나. 메모리 컴팩션

플랫폼에서 동적으로 사용하는 메모리를 할당/해제 할 때 메모리 단편화를 줄이기 위해 메모리 압축(Compaction)을 하도록 되어 있다. 이 기능은 호출하는 것이 아니라 플랫폼에서 자동으로 수행되어 진다.

다. 자바 가비지 컬렉션

플랫폼은 자바 언어 문맥에 따라 가비지 컬렉션(Garbage Collection)을 지원하도록 되어 있다. 즉, 콘텐츠 개발자는 메모리 관리를 신경 쓰지 않아도 되므로 매우 편리한 기능이지만, 빠른 메모리 확보를 위해 개발자가 가비지 컬렉션을 호출할 수도 있다.

라. 자바 스택

플랫폼은 자바 응용 프로그램별로 스택(Stack)을 할당/해제할 수 있어야 하며, 각 응용 프로그램별 스택의 크기를 동적으로 변화시킬 수 있도록 하였다. 자바 응용 프로그램이 메모리 한계를 넘는 스택 할당 요청을 했을 경우 플랫폼은 예외상황(Exception)을 응용 프로그램에 전달해야 하며, 예외상황 발생 후 플랫폼은 정상 동작하도록 되어 있다.

마. 공유 메모리 지원

응용 프로그램들이 사용하는 메모리는 서로 독립적이어야 하고, 플랫폼은 응용 프로그램 간에 공유할 수 있는 메모리를 지원하도록 하였다. 단, C 언어로 응용 프로그램을 개발할 경우는 포인터를 사용하는 언어적 특성 때문에 응용 프로그램들 간 메모리의 독립성이 유지되지 않으므로 예외로 하였다. 플랫폼은 공유하는 모든 응용 프로그램이 종료될 경우 자동으로 공유 메모리를 해제하도록 되어 있다.

(11) 응용 프로그램 관리

응용 프로그램 생명주기 및 다운로드 관리에 대해 다음의 기능을 플랫폼에서 제공하도록 한다. 별도의 API로 규격을 정하지 않지만 다음의 기능을 제공하여 응용 프로그램을 관리한다.

가. 관리 기능

1) 응용 프로그램 수행시 날짜 제한, 회수 제한 설정에 따라 기동 여부를 판단해야 한다.

2) 응용 프로그램 설치/삭제 기능을 제공해야 한다.

3) 응용 프로그램 정지 기능을 제공할 수 있다. 정지 기능은 실행 프로그램만 삭제하고 관련 데이터 파일을 남겨 두어 추후 다시 프로그램을 설치하면 이전 데이터를 활용할 수 있도록 하는 기능이다.

4) API 추가/갱신 기능을 제공할 수 있다.

5) 응용 프로그램 강제 종료 기능을 제공해야 한다.

나. 응용 프로그램 다운로드 기능

플랫폼은 응용 프로그램을 다운로드받는 기능을 지원하고, 다운로드 중 오류가 발생할 경우 초기 상태로 복구해야 한다. 무선망뿐만 아니라 시리얼 인터페이스를 통한 다운로드기능이 지원하도록 되어 있다. 시리얼 인터페이스를 통한 다운로드기능의 경우 콘텐츠 개발자들이 비싼 통신료를 지불하지 않고도 개발과정에서는 유선망을 통해 다운로드하여 개발할 수 있으므로 편리하게 사용될 수 있다.

(12) 다국어 지원

유니코드를 지원하며, 지역정보 설정을 지원하여 지역에 적합한 언어와 기호를 표시할 수 있도록 지원한다. 그리고 유니코드를 확장하여 지역 언어에 따라 추가될 수 있는 그래픽 문자를 정의한다.

가. 유니코드 지원

플랫폼은 자바 응용 프로그램을 위해 유니코드를 지원해야 하며, 입출력시 문자열은 지역 특성에 맞게 해당되는 문자 코드로 변환하도록 되어 있다. 한국의 경우는 유니코드 문자열과 EUCClientKR 문자셋(Character Set) 문자열로 상호 변환해야

한다.

나. 로케일 지원

플랫폼은 C 응용 프로그램에 대해 지역정보에 따라 참조하여 지원하는 문자셋으로 인식해야 한다. 한국의 경우는 EUCClientKR 문자셋을 이용하도록 한다. C 응용 프로그램에서 유니코드를 지역 문자셋으로 상호 변환하는 API를 제공한다.

다. 확장된 유니코드

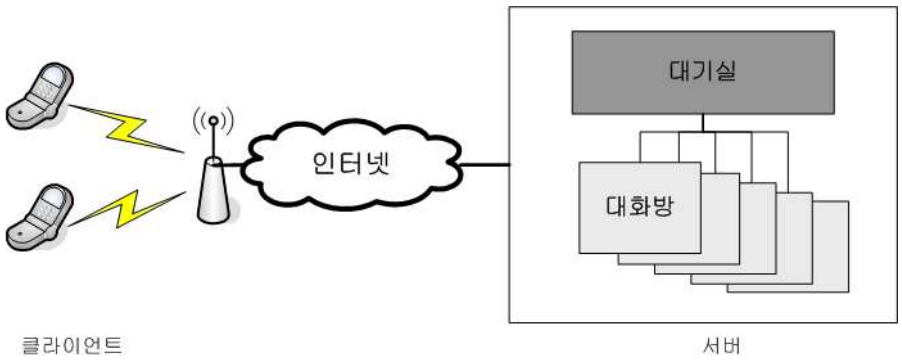
EUCClientKR 문자셋에는 유니코드에 대응되지 않는 그래픽 문자가 있으며, 이를 지원하기 위해서 유니코드 사양에서 Private Use(0xE000-0xF8FFF) 영역을 사용하는 확장된 유니코드를 사용할 수 있다[4],[6].

제 3 장 시스템의 설계

3.1 문자 대화 서비스의 설계

3.1.1 문자 대화 서비스의 구성

본 논문에서 구현할 문자 전송 서비스는 각 클라이언트인 휴대폰에서 문자 대화 프로그램을 다운로드 한 후 무선 인터넷을 통해 서버로 접속하는 형태를 가진다.



<그림 3-1> 문자 전송 서비스 구성도

각 클라이언트인 휴대폰은 문자 대화 프로그램을 다운로드한 후 무선 인터넷을 통해 서버로 접속한다. <그림 3-1>은 문자 전송 서비스의 구성도를 보여준다.

서버로 접속한 후 사용자들은 별명을 사용하여 서로 인식하게 된다. 각 사용자들은 서로 메시지를 전달할 수 있고, 또한 특정 상대에게만 메시지를 전달할 수도 있다. 그리고 문자 전송 중 현재 대기실에 있는 사람들의 목록을 볼 수 있거나, 문자 전송 중 특정 사람을 불러들여 대화를 하고, 모든 사용자 중 원하는 사용자를 검색하

여 현재 상태를 알 수 있다. 또한 문자 전송 중 대화방을 만든 사람은 대화중인 사람의 상태를 관리하는 방장의 권한을 가지고, 또한 임의의 사용자를 퇴장시킬 수 있다.

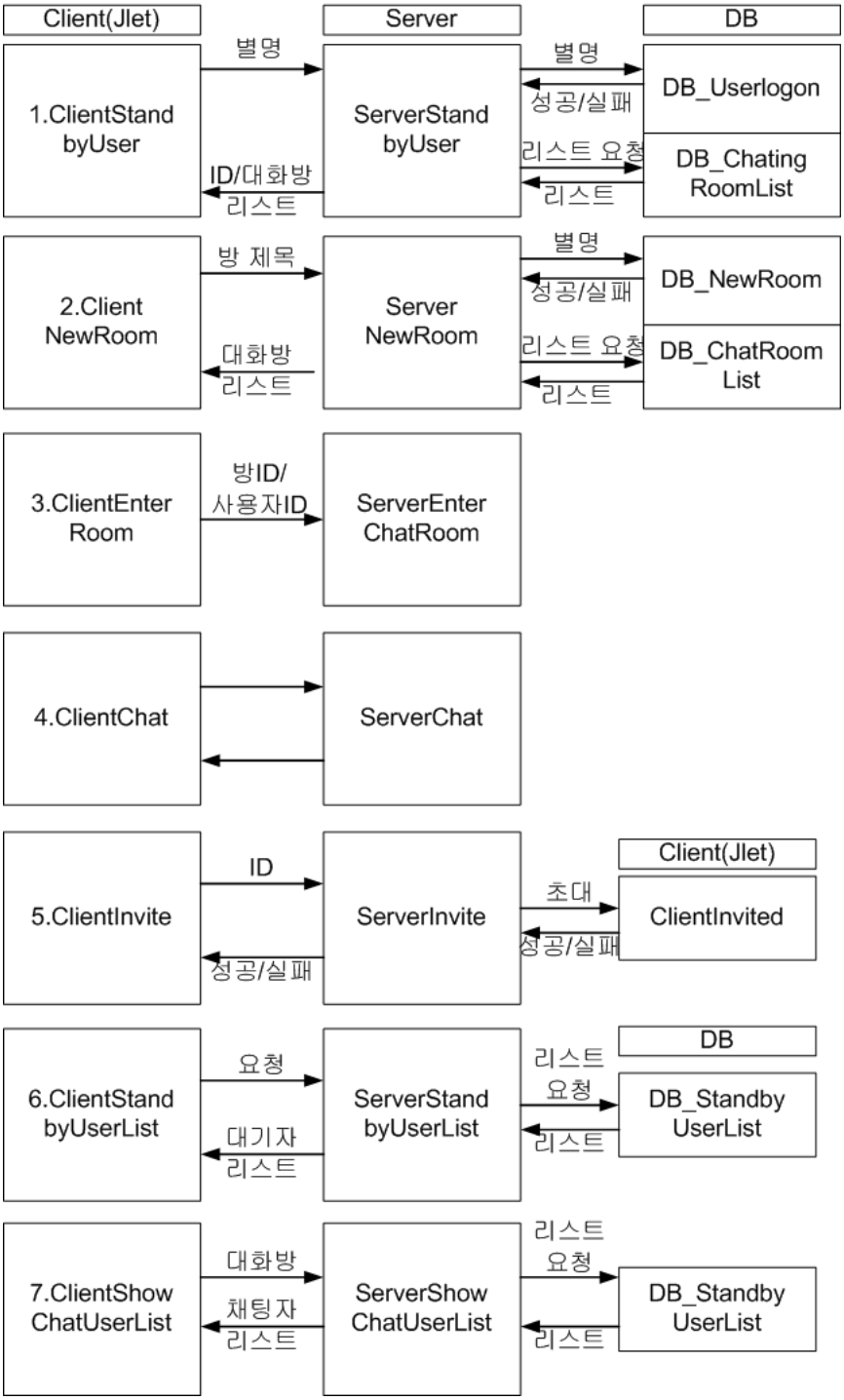
대화방은 임의로 생성되며 생성되는 개수 제한은 없다. 또한 입장할 수 있는 권한을 가지며 그 권한을 가진 사람만이 입장이 가능하다. 대화방은 대화방의 고유 이름을 가지며 한 대화방에 동시에 입장할 수 있는 인원은 제한되어 있으며 방을 개설하는 사람이 설정할 수 있다. 사용자가 문자 전송을 위해 서비스에 접속하면 대기실로 이동한다. 대기실에서는 대기 중인 사용자의 목록이나 대화중이거나 대기중인 사용자들의 상태를 볼 수 있고 대기실에 접속한 사용자는 대화방을 개설할 수 있다.

서버 측에서는 전체 사용자를 알 수 있으며, 전체 사용자에게 메시지를 보낼 수 있는 기능을 제공한다. 또한 동시 접속자 수에는 제한이 없어야 한다. 처리할 수 있는 한계를 넘으면 새로운 서버로 전달할 수 있어야 한다.

3.1.2 문자 대화 서비스의 설계

문자 대화 서비스는 클라이언트, 서버, 데이터베이스로 구성된다. 클라이언트는 콘텐츠를 다운받은 후 프로그램을 실행하여 서버에 접속한 후 ID를 생성하여 사용자들간에 문자 전송을 할 수 있는 서비스이다.

서비스 흐름의 순서는 <그림 3-2>에 나타낸다.



<그림 3-2> 문자 대화 서비스 블록 다이어그램

(1) CStandByUser 클래스

클라이언트의 CClientStandByUser는 사용자로부터 별명을 받아서 서버에 로그인 한다. 그 후 서버로부터 리스트와 자신의 사용자 ID를 받아 화면에 대화방 리스트를 보여준다.

서버의 CServerStandByUser는 Jlet 프로그램으로부터 별명을 받아서 대화방을 생성하고 DB(Data Base)에 로그를 남긴다. DB로부터 대화방 목록을 가져온 후 접속되어 있는 클라이언트마다 ID를 부여하고 ID와 대화방 목록을 내려 보내준다.

DB의 DB_Userlogon은 별명을 대기자 리스트에 등록하고 DB_ChatingRoomList는 현재 개설되어 있는 대화방 리스트를 서버에게 알려준다.

(2) CNewRoom 클래스

클라이언트의 CClientNewRoom은 사용자로부터 방 제목, 최대 인원 수, 비밀번호를 입력받아 서버에 대화방 생성을 요청하고, 서버로부터 대화방 리스트를 받아와 클라이언트 화면에 대화방 리스트를 그려준다.

서버의 CServerNewRoom은 Jlet 프로그램으로부터 별명을 받아서 대화방을 생성한다. 대화방 생성시 대화방 ID는 순차적으로 증가한다. 대화방을 생성하고 DB에 등록한 후 대화방 목록을 Jlet에 전달한다.

(3) CEnterRoom 클래스

클라이언트의 CClientEnterRoom은 사용자로부터 대화방을 선택받고 서버에 대화할 대화방 ID, 사용자의 ID를 전달한다. 만약 대화방에 비밀번호가 있으면 비밀번호를 확인한다. 그 후 서버로부터 성공/실패 여부를 확인받고 문자 전송 화면을 그려준다.

서버의 CServerEnterRoom은 Jlet 프로그램으로부터 사용자 ID와 대화방의 ID를 받고 사용자를 창에 추가 시킨다.

(4) CChat 클래스

클라이언트의 CClientChat은 사용자로부터 메시지를 받아 대화방 전체클라이언트에게 보내고 서버로부터 대화 메시지를 화면에 그려준다.

서버의 CServerChat은 Jlet 프로그램으로부터 대화방의 메시지를 받아서 대화방 전체에게 전달한다.

(5) CInvite 클래스

클라이언트의 CClientInvite는 대기 중에 있는 사용자에게 대화를 요청한다. 서버의 CServerInvite는 Jlet 프로그램으로부터 별명을 받아 해당 사용자에게 대화 요청을 알리고 성공/실패를 요청한 사용자에게 알려준다. 클라이언트의 CClientInvited는 다른 사용자로부터 대화 요청을 받아서 화면에 표시한다(CClientInvite와 다름).

(6) CStandbyUserList 클래스

클라이언트의 CClientStandbyUserList는 대기실에 있는 사용자의 리스트를 서버에게 요청하여 사용자에게 알려준다. 서버의 CServerStandby UserList는 Jlet 프로그램으로부터 시그널을 받아 DB에 조회하여 대기 중인 사용자 리스트를 돌려준다. DB의 DB_StandbyUserList는 대기실에 있는 사용자 리스트를 서버에게 알려준다.

(7) CShowChatList 클래스

클라이언트의 CClientShowChatList는 현재 대화중인 방의 사용자

리스트를 얻는다. 서버의 CServerShowChatList는 Jlet 프로그램으로부터 대화방 ID를 받아 대화방에 존재하는 사용자의 리스트를 클라이언트에게 알려준다.

3.2 상용어구 기능의 설계

3.2.1 상용어구 기능의 개요

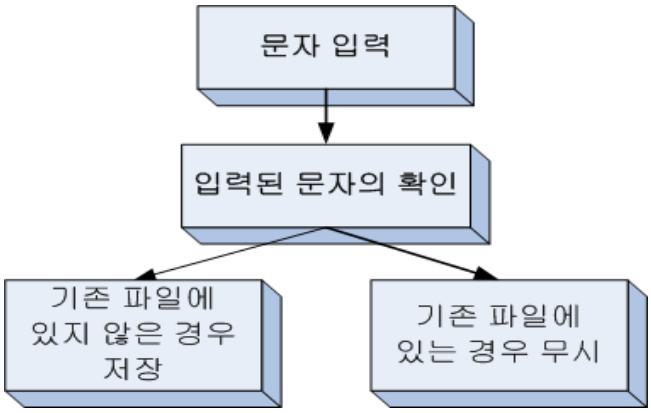
상용어구 기능은 문자 대화 서비스를 보다 쉽게 이용하기 위하여 설계되었다. 현재 각 통신사들마다 휴대폰의 문자 입력 방식에 조금씩의 차이가 있다. 자음 입력방식에는 차이가 없으나 모음 입력 방식에 있어서 동일키를 여러 번 눌러 자음을 선택하는 방법과 다른 키와 조합하여 모음을 선택하는 방법이 있다. 이처럼 휴대폰에 따른 입력방식의 차이로 인하여 사용자가 금방 적응하기 힘든 경우가 있다.

또한 휴대폰 자판에 익숙하지 못하여 문자 입력이 늦을 경우 대화 상대방 쪽에서 답답함을 느끼는 경우도 있다. 특히 여러 명이 대화를 하는 경우 문자 입력이 늦어 서로간의 대화가 늦어지는 일이 발생하는 경우가 있다. 따라서 상용어구 기능을 구현함으로서 사용자가 보다 편리하고 빠르게 문자 전송 서비스에 적응할 수 있다.

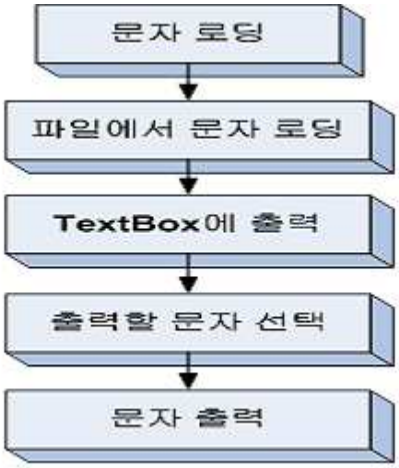
3.2.2 상용어구 기능의 동작

상용어구 기능은 문자 대화 서비스의 클라이언트 프로그램에서 동작한다.

<그림 3-3>에서 보듯이 클라이언트 측에서 문자를 입력할 경우 그 문자의 존재 여부에 따라 파일에 저장을 하게 된다.



<그림 3-3> 상용어구의 저장



<그림 3-4> 상용어구 로딩 흐름도

<그림 3-4>는 상용어구의 로딩 흐름도이다. 상용어구 로딩의 경우에는 특수키인 *키와 자음키가 사용이 된다. *키 다음에 자음키가 입력될 경우 그 자음으로 시작하는 문자를 상용어구 선택 창에 보여주게 되며 사용자가 자신이 원하는 문자의 번호를 선택하면 입력이 된다.

상용어구 파일의 크기는 단말기의 특성상 많은 데이터를 저장할 수 없으므로 자동으로 파일내의 데이터를 삭제하는 알고리즘을 필요로 한다. 본 논문에서는 페이징 교체 알고리즘의 하나인 LRU(Least Recently Used : 최소 최근 사용) 알고리즘을 사용하여 상용어구 파일의 크기를 관리한다.

제 4 장 시스템의 구현 및 실험

4.1 문자 전송 서버의 구현

서버는 클라이언트로부터 별명을 받게 되면 그 별명을 확인하여 대화방 리스트를 클라이언트에게 넘겨준다. 또한 클라이언트가 대기자 리스트나 현재 대화방에 접속 중인 사용자 리스트를 요구하면 넘겨주는 기능을 한다.

4.1.1 서버 사용 클래스의 구현

서버를 구성하는 주요 클래스들은 다음과 같다.

(1) CServerServer 클래스

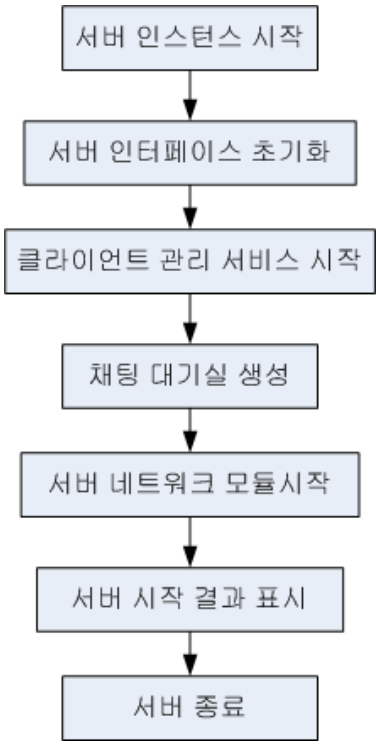
NetworkStarter를 상속받으며 NetworkStarter를 구현한 클래스이다.

CServerServer 클래스는 서버 프로그램의 시작점이 되는 클래스이고, 이 클래스는 서버 소켓을 생성하고 클라이언트의 접속을 기다린다. 그 후 클라이언트가 접속하면 생성된 소켓을 가지고 CServerUnit 클래스를 생성한다. CServerUnit 클래스의 startService()를 호출하여 입출력 스트림을 생성하고 상속받은 NetworkInterface의 receiveFrame() 메서드(method)를 수행한다.

상속받은 클래스의 메서드 receiveFrame()이 호출되면 CServerServer 클래스의 processReadRequest()가 호출된다. 이 메서드는 CServerUnit 클래스의 receive() 메서드를 호출하고, receive() 메서드는 입력 스트림을 통해서 데이터를 읽기 시작한다. 첫 바이트는 버전이기 때문에 버전이 맞지 않으면 더 이상 읽지 않

고 다시 NetworkStarter의 receiveFrame()을 호출한다. 만약 버전이 맞는다면 NetworkInterface의 sendFrame()을 호출한다. 이 메서드를 호출하면 다시 CServerServer 클래스의 processWriteRequest()를 호출하고 마지막으로 CServerUnitSender의 send() 메서드가 호출되어 프레임의 다음 바이트를 처리하게 된다.

<그림 4-1>은 CServerServer 클래스의 흐름도를 나타낸다.

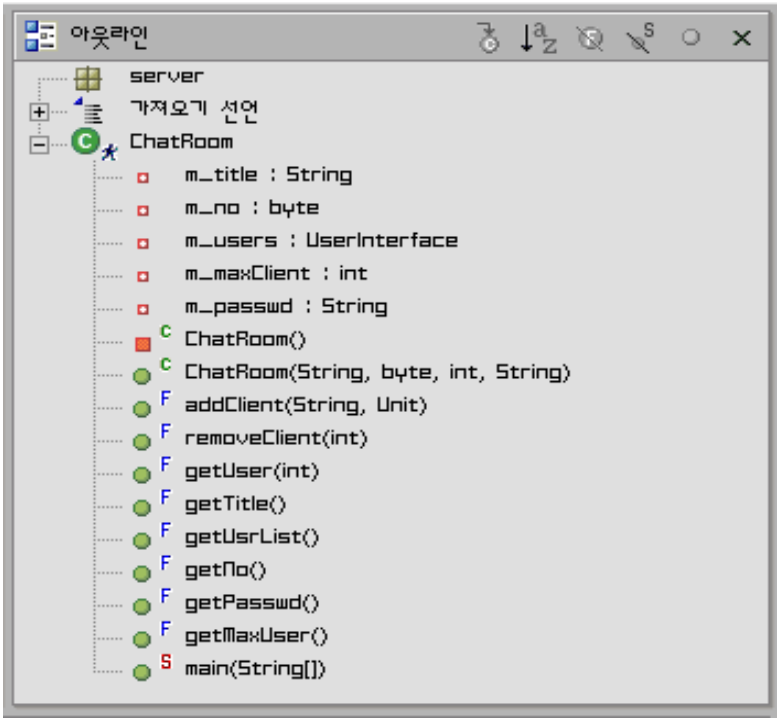


<그림 4-1> CServerServer 클래스의 흐름도

CServerServer 클래스는 서버 인스턴트 시작 클래스로서 최대 대화방 개수와 사용할 TCP(Transmission Control Protocol) 포트를 설정한 후 서버 인스턴트를 시작한다. 인스턴스 시작 후 대기실 생성 기능을 가지고 서버 네트워크 모듈을 시작한다.

(2) CServerChatRoom 클래스

CServerChatRoom 클래스는 대화방을 표현한 클래스이다. <그림 4-2>는 CServerChatRoom 클래스의 UML(Unified Modeling Language) 다이어그램을 나타낸다.



<그림 4-2> CServerChatRoom 클래스의 UML 다이어그램

서버가 관리하는 CServerChatRoom과 클라이언트가 관리해야 하는 CClientChatRoom과는 많은 차이가 있다. 또 서버는 대화방과 대기실을 같은 객체인 CServerChatRoom을 써서 관리한다. 그러나 대기실의 경우에는 ID를 99번으로 주어 구별하였다.

이 클래스의 맴버 변수는 대화방의 최대 사용자 수를 나타내는 m_maxClient와 대화방의 번호를 가지고 있는 m_no, 대화방의 비밀

번호를 가지고 있는 m_passwd, 또 방 제목을 가지고 있는 m_title 과 대화방 사용자를 관리해 주는 인터페이스인 m_users가 있다.

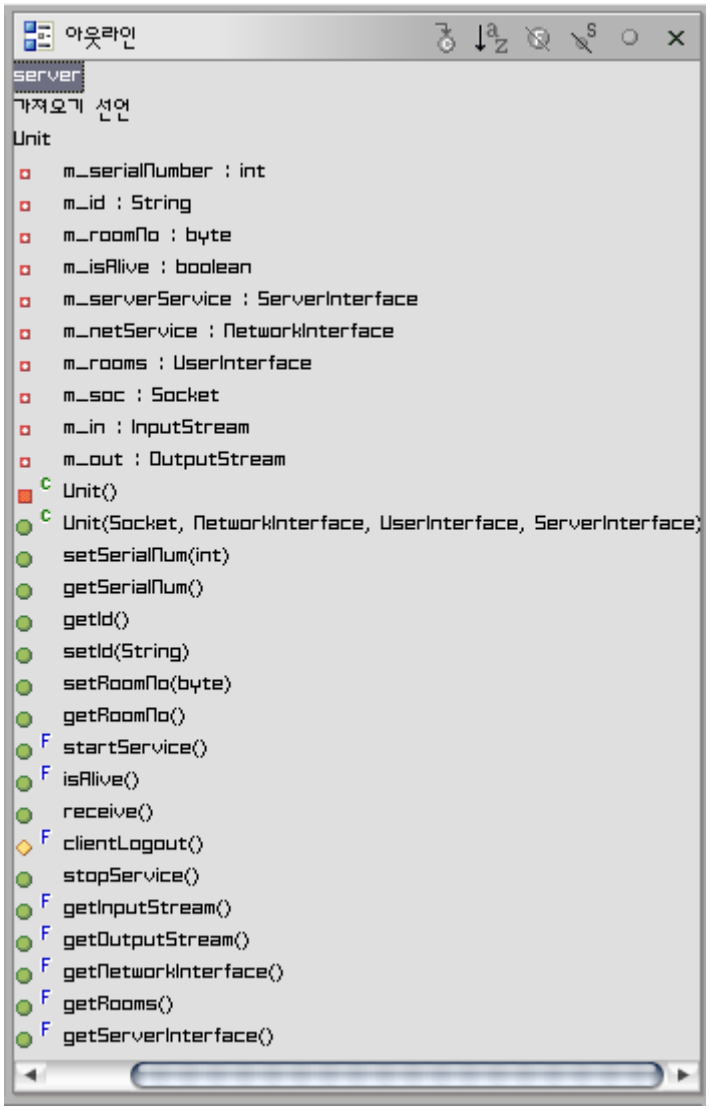
이 클래스의 메서드는 대화방에 사용자를 추가하는 addClient()와 사용자를 리턴 받는 getUser(), 그리고 사용자를 얻어오는 메서드 getUsetList()가 있다. 그 외에 멤버변수 값을 얻어오는 getNo(), getMaxUser(), getPassword()로 구성된다. 주요 기능은 클라이언트 관리 서비스를 시작하여 대화 사용자를 추가하고, 대화방 주제 또는 방 내에 존재하는 사용자 리스트를 얻거나 대화방 번호, 암호를 얻어 오는데 사용된다.

(3) CServerUnit 클래스

이 클래스는 클라이언트가 서버에 접속하면 생성되며 클라이언트와 통신하는 소켓을 통하여 데이터를 주고받는 모든 기능을 가지고 있다. 이 클래스는 서버 소켓의 accept()의 리턴 값으로 받은 소켓과 모든 서버의 수행을 관장하는 클래스 NetworkStarter, 대화방 관리 인터페이스와, ServerInterface를 가지고 생성된다. 이 클래스의 UML 다이어그램은 <그림 4-3>과 같다.

이 클래스의 멤버 변수 m_serialNumber은 클라이언트의 ID를 나타내고, m_id는 클라이언트의 ID를 String으로 바꾼 것이다. 그리고 m_roomNo는 현재 클라이언트가 있는 방 번호를 나타낸다. m_in, m_out, m_soc는 네트워크과 관련된 멤버 변수이고, m_isAlive는 네트워크가 살아있는지를 나타내는 멤버 변수이다.

이 클래스의 메서드는 멤버 변수의 값을 얻어오거나 설정하는 메서드와 isAlive() 메서드와 receive()메서드가 있다. isAlive()는 네트워크가 살아있는지를 나타내는 메서드이고, receive()는 클라이언트로부터 한 프레임을 받는 메서드이다.



<그림 4-3> CServerUnit 클래스 UML 다이어그램

이 메서드는 버전을 검사하고 버전이 맞으면, CServerUnitSender의 send() 메서드를 호출하여 전달받은 데이터를 명령코드에 맞게 처리한 후 다시 receive()가 호출되도록 한다. receive()는 클라이언트에서 전송하는 스트림의 지연시간이 TimeOut에서 지정한 시간보

다 길 경우 지연시간이 긴 패킷은 다음에 검사하고, 다음 스트림의 지연 시간을 줄이기 위해 다음 스트림을 바로 처리한다.

그 외에 버전을 받고, 메시지코드에 맞게 적절한 수행을 하는 클래스인 CServerUnitSender 클래스와 서버 클라이언트간의 명령어코드를 정의해 놓은 CServerProtocolInterface 클래스, 서버와 클라이언트간의 에러를 규정한 CServerErrorMessage 클래스와 이 클래스간의 호환을 위해 사용되는 CServerServerInterface 클래스로 구성된다.

4.2 문자 전송 클라이언트의 구현

클라이언트는 사용자가 문자 전송 서비스에 접속하면 별명을 생성하게 하고 대화방 리스트를 넘겨받는 기능을 한다. 그리고 대화방에 참여하거나 대화방을 생성하여 대기자를 초대할 수 있는 기능을 제공한다.

클라이언트는 서버와 마찬가지로 여러 개의 클래스들로 구성된다. 각 클래스들은 다음과 같다.

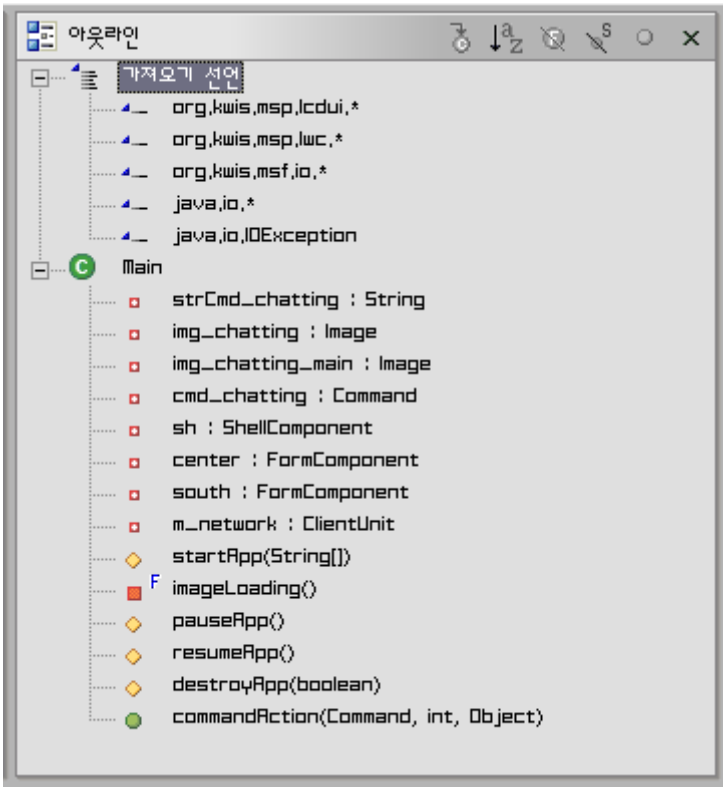
4.2.1 클라이언트 사용 클래스의 구현

클라이언트는 클라이언트를 구동하는 CClientMain 클래스와 대화방을 생성하기 위한 CClientNewRoom 클래스, 사용자의 리스트를 얻어오는 CClientStandbyUserList, CClientShowChatUserList 등으로 구성된다.

(1) CClientMain 클래스

클라이언트 프로그램의 CClientMain 클래스는 프로그램 첫 화면

을 그려주는 클래스이다. <그림 4-4>는 CClientMain 클래스의 UML 다이어그램을 나타낸다.



<그림 4-4> CClientMain 클래스 UML 다이어그램

멤버 변수를 살펴보면, center는 LCD(Liquid Crystal Display)의 작업 컴포넌트 영역에 삽입될 FormComponent이다. 이 컴포넌트에 ImageComponent를 추가시켜 메인 이미지가 출력되도록 하였다. 멤버 변수 south는 cmdBar가 추가되었으며 cmdBar에는 “대화”를 실행시키기 위한 Command 클래스가 있다. 사용자가 대화가 선택된 상태에서 EventQueue.FIRE를 발생시키면, 대화 로그인을 하고 다음 화면으로 넘어간다. 네트워크를 초기화하기 위해 멤버 변수 m_network를 갖는다. 이 멤버 변수의 타입은 CClientClientUnit이

다.

이 클래스의 메서드는 Jlet을 실행시키기 위한 4가지 메서드 이외에 imageLoading()과 cmdBar의 이벤트를 처리하는 Command Action() 메서드가 있다. imageLoading() 메서드는 화면에 뿌려질 그림을 읽어 들이는 메서드이다.

```
public void commandAction(Command c, int type, Object obj)
{
  String cmd = c.getString();

  if (cmd.equalsIgnoreCase(strCmd_chatting) && type == 2)
  {
    StandbyUser _enter_chatting_waiting_room = new
      StandbyUser(sh, this.m_network);
  }
}

}
```

<그림 4-5> 입력 처리

<그림 4-5>는 CClientMain 클래스에서 사용자의 입력을 처리하는 부분이다. commandAction() 메서드는 네트워크를 초기화하고 화면을 이동시키는 역할을 한다. 사용자의 입력을 처리하기 위해서 파라미터로 전달된 Command와 Event 타입을 이용하여 구별한다. Event 타입은 type 값이 1이면 cmdBar가 이동되었음을 알려주고, type 값이 2 이면 cmdBar가 선택되었음을 나타낸다.

프로그램을 실행시키면 <그림 4-6>의 (a) 화면이 나타난다. 여기서 단말기의 키패드의 방향키를 움직이면 (b) 화면처럼 대화 메뉴가 선택이 된다.

(a) 실행 (b) 대화메뉴

<그림 4-6> CClientMain 클래스 실행화면

(2) CClientStandbyUser 클래스

CClientStandbyUser 클래스는 대기실에 들어가기 위하여 사용자로부터 대화명을 입력받고, 서버에 전달하여 결과에 따라 대기실에 입장하는 클래스이다. Runable 인터페이스와 ActionListener 인터페

이스로 구성된다. 화면은 작업 컴포넌트 영역과 명령 컴포넌트 영역으로 구분되며, 작업 컴포넌트 영역에는 대화명을 입력받는 역할을 하고 명령 컴포넌트 영역은 “취소”, “입장”, “도움말” 메뉴 버튼이 존재한다.

이 클래스의 메서드는 5개가 있다. Load()는 화면에 화면을 그려주는 메서드이고, imageLoading()은 사용될 이미지를 로드하며, run() 메서드는 에러가 발생하면 화면에 에러를 보여준다.

```
...
btn_cancel.setActionListener(this, strBtn_cancel);
btn_enter.setActionListener(this, strBtn_enter);
btn_hangul.setActionListener(this, strBtn_hangul);
...

...
public void action(Component c, Object o)
  {
    String cmd = (String)o;
    //취소 버튼이 눌러지면
     if (cmd.equalsIgnoreCase(strBtn_cancel))
    {
       ...
    }
    //확인 버튼이 눌러지면
     else if (cmd.equalsIgnoreCase(strBtn_enter))
    {
       ...
    }
```

<그림 4-7> 버튼 액션 처리

<그림 4-7>은 action() 메서드에서 버튼을 처리하는 부분이다. 사용자가 “취소”, “입장”, “도움말” 버튼을 눌렀을 때에 이를 처리하는 이벤트 핸들러 메서드이다. 세 개의 버튼을 모두 누르면 action() 메서드를 호출하는데 각각을 구별하기 위하여 action() 메서드의 두

번째 파라미터 오브젝트를 사용한다. “취소”를 누르면 tfNickname의 모든 String을 “ ”로 초기화시키며, “도움말”을 누르면 도움말이 보이는 창으로 이동한다.

```
boolean ret = m_network.startService("127.0.0.1", 20000, this.m_network.Id);
```

<그림 4-8> 입력 스레드 수행

“입장” 버튼을 누르면, 먼저 대화명이 정확한지를 검사한 후 네트워크에서 들어오는 모든 입력을 받는 스레드를 수행시키기 위해 <그림 4-8>을 수행한다.

만약 네트워크를 사용할 수 없다거나 대화명이 입력되지 않았다거나 대화명이 입력되지 않았다거나 대화명이 이미 존재하는 경우에는 화면에 에러창을 출력한다.

loadStandbyRoom() 메서드는 서버에 대화명을 전달한 후, 서버로 응답 메시지를 받은 ListenThread가호출하는 메서드이다. 서버로부터 성공을 받으면 대기실로 이동하고 실패를 받으면 실패 메시지를 사용자에게 보여준다.

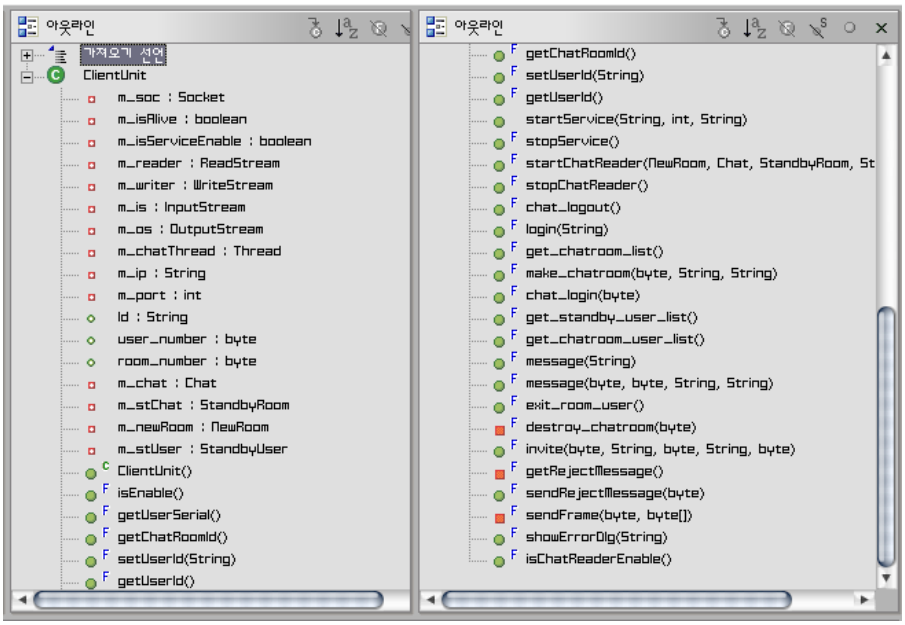
(a) 대화명 입력 **(b) 입력에러** **(c) 도움말**

<그림 4-9> CClientStandbyUser 클래스 실행화면

<그림 4-9>는 CClientStandbyUser 클래스의 실행화면이다. CClientMain 클래스의 실행화면에서 대화를 선택하면 (a) 화면이 나타난다. 이때 대화명 입력란에 별명을 입력하면 된다. 만약 대화명을 입력하지 않거나 동일한 대화명이 있을 경우 (b)와 같은 에러화면을 출력한다. (c)는 도움말을 선택하였을 때 나타나는 도움말 화면이다.

(3) CClientClientUnit 클래스

CClientMain 클래스의 startApp() 메서드가 호출되자마자 생성하는 것이 CClientClientUnit 클래스로서 네트워크와 관련된 모든 기능을 수행하는 클래스이다. 서버로부터 들어오는 입력을 받아 의미를 파싱하고, 적당한 일을 수행할 수 있게 하며, 사용자의 요구에 적절한 데이터가 네트워크를 통해 서버에 전달하는 기능을 수행한다.



<그림 4-10> CClientClient 클래스 UML 다이어그램

<그림 4-10>은 CClientClient 클래스 UML 다이어그램이다.

멤버 변수 ID는 사용자로부터 입력받은 별명을 나타낸다. 변수 room_number는 클라이언트가 대화중일 경우 현재 속해 있는 대화방을 구별하는 식별자이다. user_number는 서버가 내려주는 클라이언트 구분자이다. 별명과 더불어 user_number가 자신을 나타내는 ID이며 user_number값을 가지고 사용자를 구별한다.

CClientClientUnit 객체는 대화 메시지를 전달받으면 Chat 클래스로 전달하여 처리한다. Chat 클래스로부터 서버에 전달할 메시지를 받아와서 서버에 전달하는 기능도 수행한다.

CClientClientUnit의 메서드 중 생성자를 제외한 가장 먼저 생성되는 메서드는 isEnable()이다. 네트워크가 초기화되어 네트워크를 사용할 수 있을지를 알아내는 메서드이다. 두 번째로 사용되는 메서드는 startService() 이다. <그림 4-11>의 메서드는 네트워크를 초기화하며, 소켓을 생성하고, 입출력 스트림을 초기화한다.

```
int ret = Network.connect();

  if (ret != -1)
  {
   m_soc = URL.find("socket://"+m_ip.trim()+":"+String.valueOf(m_port));
   m_is = m_soc.getInputStream();
   m_os = m_soc.getOutputStream();
   m_reader = new ReadStream(m_is);
   m_writer = new WriteStream(m_os);
  }
  else
  {
   m_isServiceEnable = false;
   return false;
  }
```

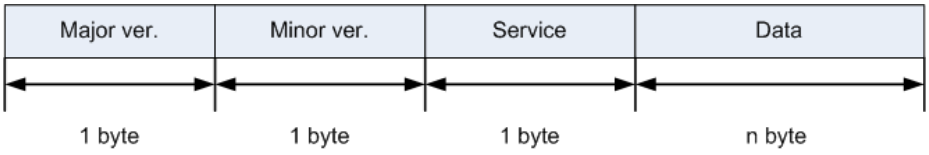
<그림 4-11> 초기화 실행

<그림 4-12>는 사용되는 ReadStream과 WriterStream을 나타낸다.

```
public final String readSrting() throws IOException
{
   int size = readShort();
   if (Size<0) throw new EOFException();
   byte[] content = new byte[size];
   read(content, 0, size);
   return new String(Content);
}
   {
      byte[] = str.getBytes();
      int size =content.length;
      writeShort((short)size);
      write(content, 0, size);
   }
```

<그림 4-12> ReadStream, WriteStream

서버와 프로토콜은 ProtocolInterface 인터페이스에 정의되어 있으며, 메시지의 규격은 <그림 4-13>과 같다.



<그림 4-13> 메시지 규격

첫 바이트는 Major version을 보내고, 그 다음 바이트는 Minor version을 보내고, 그 다음 바이트는 Service 코드를 보내며, 마지막으로 Service 코드에 따라 가변적으로 데이터를 전송하도록 구현하

였다.

```
private final void sendFrame(byte service, byte body[]) throws IOException
 {
  m_writer.writeByte(MAJOR_VER);
  m_writer.writeByte(MINOR_VER);
  m_writer.writeByte(service);

  if (body.length > 0)
   m_writer.write(body, 0, body.length);

  m_writer.flush();
 }
```

<그림 4-14> sendFrame()

메시지를 서버로 전송하기 위해서 <그림 4-14>의 메서드 sendFrame()을 사용하는데, 이 메서드는 파라미터로 Service 코드와 이어질 데이터를 갖는다.

(4) CClientStandbyRoom 클래스

이 클래스는 대기실을 구현한 클래스이다. 대기실에서 필요한 대화방 목록을 가지고 있으며 대화방과 관련되는 메뉴를 가지고 있다. 대화방 목록은 작업 컴포넌트 영역에 존재하고, 메뉴는 명령 컴포넌트 영역에 존재한다. 대기실에서 제공하는 메뉴는 “대화방 참여”, “대화방 개설”, “이전 항목”, “다음 항목”, “종료”가 있다.

이 클래스의 메서드는 화면을 구성하는 load(), imageLoading()이 있다. 이벤트는 두 곳에서 발생할 수 있는데 대화창 리스트에서 선택된 목록이 바뀔 때와, CommandBarComponent에서 메뉴가 바뀔 때이다. <그림 4-15>은 action() 메서드의 코드이다.

```
public void action(Component c, Object o)
 {
  if (m_network.isEnable())
  {
   ListComponent list = (ListComponent)o;
   //리스트에 선택된 인덱스 얻기
   int selectedIdx = list.getSelectedIndex();
   //실제 선택된 채팅정보 얻기
   ChatRoom room = (ChatRoom)Buf_waitting_room.elementAt(selectedIdx);
   SelectedKey = room.getKey();
   SelectedTitle = room.getTitle();
   SelectedPasswd = room.getPasswd();
   SelectedInvite = room.getInviteValue();
   IsOpen = room.isOpen();
  }
 }
```

<그림 4-15> action()

action() 메서드는 대화창 목록이 바뀌었을 때 호출되는 메서드로서 이 메서드가 호출되면 SelectedXXX같이 현재 선택된 방의 정보를 담고 있는 값이 모두 새로운 값으로 대치된다.

<그림 4-16>은 commandAction() 메서드의 코드를 나타낸다. commandAction() 메서드는 CommandBarComponent의 메뉴가 변경되었을 경우 호출되는 메서드다. 파라미터 타입에 따라 메뉴를 이동하였을 때와 메뉴를 선택하였을 때를 구별할 수 있다. 메뉴를 선택하였을 때 CClientClientUnit를 통하여 해당 정보를 서버에 전달하고 결과를 받는다.

```
public void commandAction(Command c, int type, Object obj)
 {
  if (m_network.isEnable())
  {
   String cmd = c.getString();

   if (type == 2)
   {
    if (cmd.equalsIgnoreCase(strCmd_before))
          ...
    else if (cmd.equalsIgnoreCase(strCmd_next))
          ...
    else if (cmd.equalsIgnoreCase(strCmd_enter_chatting_room))
          ...
    else if (cmd.equalsIgnoreCase(strCmd_make_room))
         ...
    else if (cmd.equalsIgnoreCase(strCmd_exit))
          ...
   }
  }
 }
```

<그림 4-16> CClientStandbyRoom 클래스 commandAction()

(5) CClientChatRoom 클래스

CClientChatRoom 클래스는 대화방을 표현하는 클래스로서 대화방이 가지는 모든 정보를 멤버 변수가 가지고 있다. 공개, 비공개를 나타내는 정적 변수인 NOT_OPEN, Open을 가지고 있고, 멤버 변수 m_isInviteRoom은 대화방으로부터 초대 요정에 대한 정보를 가지고 있다. m_max는 방 인원의 최대 인원수를 저장하고 있다. m_passwd는 방의 비밀번호를 가지고 있으며, m_strKey는 m_key값을 문자열로 저장하기 위한 변수며, m_title는 방 제목을 담고 있다.

(6) CClientNewRoom 클래스

CClientNewRoom 클래스는 CClientStandbyUser 클래스에서 새로운 “대화방 생성”을 선택하였을 경우 생성되는 클래스이다. 이 클래

스는 새로운 대화방을 생성하는 사용자 인터페이스를 제공하고 사용자로부터 방 생성에 필요한 값을 입력받아 서버에 전달하고 결과에 따라 화면을 이동하는 역할을 한다.

사용자로부터 방 생성에 관한 정보를 입력받기 위해 사용되는 인터페이스는 각각 세 개의 TextFieldComponent, LabelComponent, ButtonComponent이다. 멤버 변수 tf_chatting_room_name는 방 제목을 입력받는 변수이다.

이 클래스의 메서드는 action() 메서드와 이 결과를 행하는 메서드와 화면을 구성하는 메서드로 이루어진다. 메서드 Load()와 imageLoading()은 화면을 꾸미는 메서드이고 action()메서드는 버튼을 누르면 호출되어 적절한 수행을 한다. 사용자가 “취소” 버튼을 누르면 입력한 모든 것이 다 지워지고 사용자가 “생성” 버튼을 누르면 사용자가 입력한 정보를 모아 서버에 전달한다. 서버에서 결과가 도착하면 showMakeChatRoomResult()가 호출되어 실패하면 실패 다이얼로그를 출력하고 성공하면 대기실로 이동한다.

(a) 대화방 참여 **(b) 대화방 생성**

<그림 4-17> CClientStandbyRoom과 CClientNewRoom의 실행화면

<그림 4-17>은 CClientStandbyRoom 클래스와 CClientNewRoom 클래스의 실행화면이다. 별명을 생성 후 서버에 입장 후 자신이 원하는 대화방을 선택하여 하단의 CommandBar에서 (a) 화면에서 대화방 참여를 선택하면 대화방에 입장이 가능하다. (b) 화면은 대화방 참여가 아닌 대화방 생성을 선택하여 대화방을 생성하는 화면이다.

(7) CClientChat 클래스

CClientChat 클래스는 대화 화면을 구성하고 사용자로부터 대화 메시지를 받아들여 화면에 출력하는 역할을 하는 클래스이다. 대화 중 대기자 리스트 보기, 대화방 사용자 리스트 보기 등 다른 기능으로 이동할 수 있도록 하는 역할도 담당한다.

전체 화면은 작업 컴포넌트와 명령 컴포넌트 영역으로 구분하여 사용한다. 작업 컴포넌트 영역은 사용자가 입력한 메시지를 보여주기 위하여 TextBoxComponent 컴포넌트를 사용하였고, 명령 컴포넌트 영역에는 사용자가 메시지를 입력할 수 있도록 하였다.

사용자가 만약 EventQueue.SOFT1 이벤트를 발생시키면 CClientChat Menu 클래스를 명령 컴포넌트 영역에 추가하여 대기 중 사용자 리스트나 대화방 사용자 리스트를 볼 수 있는 메뉴를 제공한다.

<그림 4-18>의 메서드 load()는 화면의 컴포넌트를 구성하는 것이며, imageLoading()은 화면에 필요한 이미지를 읽어오는 메서드이다.

메서드 changeCommandMode()는 명령 컴포넌트 영역 CClientChatManu 클래스로 바꾸어 대화방 사용자 리스트와 대기실 사용자 리스트를 볼 수 있는 메뉴가 출력되도록 한다. 메서드 changeInputMode()는 이와는 반대로 화면의 명령 컴포넌트 영역을

TextFieldComponent로 바꾸어 메시지를 입력할 수 있도록 하는 메서드이다.

```
public final void changeCommandMode()
 {
  south.removeAllComponents();
 south.addComponent(m_menu);
 m_isCommandMode = true;
  }
```

<그림 4-18> ChangeCommandMode()

메서드 setMsg()는 작업 컴포넌트 영역에 메시지를 출력하는 메서드이며 setChatUserList()는 CClientStandbyUser 클래스의 잡업 컴포넌트 영역에 대기실 사용자 리스트를 출력하는 메서드이다. 메서드 setChatUserList()는 대기실 사용자 리스트를 서버에 요청하면 서버로부터 데이터를 받은 CClientUnitClient가 호출하는 메서드이며, 메서드가 호출되면 화면을 대기자 리스트가 보이는 화면으로 이동한다. showChatuserList() 메서드는 사용자가 대화방 사용자 리스트를 서버에 요청하면, 서버로부터 대화방 사용자 리스트를 받아와서 호출되는 메서드다.

(8) CClientChatMenu 클래스

이 클래스는 CClientChat 클래스의 명령 컴포넌트 영역에 추가되어 대화중인 사용자가 대기실 사용자를 초대하거나, 같은 대화방에 존재하는 사용자 리스트를 볼 수 있는 메뉴를 제공하는 클래스이다. FormComponent를 상속받았으며, commandBarComponent에 모든 Command를 포함하고 있다.

이 클래스의 메서드는 CommandBarComponent의 이벤트 처리를 하는 메서드와 전달받은 정보에 따라 처리하는 메서드가 있다.

```
public void commandAction(Command c, int type, Object obj)
{
  String  cmd=c.getString();
  if(type==2)
  {
    ...
      if(cmd.equaisIgnoreCase(strCmd_tequest))
      ...
      else if(cmd_equalsIgnoreCase(strCmd_room_people))
      ...
      else if(cmd.equalsIgnoreCase(strCmd_to_room))
      ...
  }
}
```

<그림 4-19> CClientChatMenu 클래스 commandAction()

<그림 4-19>는 이벤트를 처리하는 메서드 commandAction()이며 파라미터로 이벤트를 발생한 Command와 이벤트 타입이 전달된다. type 값을 비교하여 메뉴를 선택하였을 때와 이동하였을 때를 구분한다.

방 참여자 리스트를 보여주기 위하여 화면을 먼저 구성해 놓은 다음 서버에 대기자 리스트를 요청하고, 그 결과를 받아 메서드 setChatUserList()를 이용하여 화면에 사용자 리스트를 추가한다.

초대하기도 같은 과정으로 수행되며 setStandbyUserList() 메서드가 호출되어 목록을 다시 그려준다.

(a) 대화 화면 **(b) 대화방 참여자 보기**

<그림 4-20> CClientChat과 CClientChatMenu 클래스 실행화면

<그림 4_20>은 Lotto라는 대화방의 모습과 현재 대화방에 참여중인 사용자를 보여주는 화면이다.

(9) CClientStandbyUserList 클래스

이 클래스는 대기실에 대기 중인 사용자 목록을 화면에 보여주는 클래스이다. 화면의 작업 컴포넌트 영역에는 대기실 사용자 목록이 보여지고 명령 컴포넌트 영역에는 메뉴가 보여진다. 각 컴포넌트 영역은 ListComponent와 CommandBarComponent를 사용한다.

이 클래스의 메서드는 CClientChat 클래스와 유사하다. ListComponent의 이벤트를 처리하는 action()과 CommandBar Component의 이벤트를 처리하는 apthemcommandAction()이 있다. 또 화면을 구성하기 위해 사용되는 load()와 imageLoading()이 있다. 대기실 사용자 리스트의 추가는 이 클래스가 생성되면서 수행되지 않고 서버로부터 정보를 받아온 다음에 추가된다.

(10) CClientShowChatUserList 클래스

대화방에 있는 사용자의 목록을 보여주는 클래스이다. 이 클래스

의 구조는 CClientStandbyUserlist와 같다. 전체 화면은 대화방 사용자 목록을 보여주는 작업 컴포넌트 영역과 메뉴로 구성되어 있는 명령 컴포넌트 영역으로 구성된다. <그림 4-20>의 (b) 화면과 같이 나타난다.

(11) CClientHelp 클래스

이 클래스는 문자 전송 프로그램의 도움말을 보여주는 클래스이다. 클래스의 CClientStandbyUser에서 메뉴 “도움말”을 선택하면 <그림 4-9>의 (c) 화면으로 이동한다.

화면은 작업 컴포넌트 영역의 TextBoxComponent를 추가하고 명령 컴포넌트 영역에 ButtonComponent를 추가하였다. 버튼을 가운데 추가하기 위하여 FormComponent를 생성할 때 false로 하여 수직으로 배열하였다. 도움말을 본 다음 다시 CClientStandbyUser로 돌아가기 위해서는 “취소” 버튼을 눌러야 한다.

4.3 상용어구 기능의 구현

상용어구 기능은 미리 입력된 어구를 파일에 저장해 두었다가 로딩키(*)와 자음이 입력이 되면 파일 내에서 그 자음으로 시작되는 파일을 로딩하여 선택 창에 출력하는 기능이다.

File 클래스는 파일에 대한 읽기/쓰기 기능과 같은 기본적인 기능과 조금 더 편리하고 빠르게 사용하기 위한 스트림 기능을 지원하는 클래스이다. 파일 이름은 모두 절대경로로 되어 있다. 실재로는 플랫폼에서 허용되는 디렉터리 안에서만 파일을 이용하고 지울 수 있다. 이 디렉터리는 응용 프로그램마다 관리되므로 다른 응용 프로그램에서 사용하는 파일에 접근하기 위해서 파일을 생성시 옵션을

바꾸어야 한다.

<그림 4-21>은 파일 생성자를 나타낸다.

```
public File(String filename, int mode)
public File(String filename, int mode, int flag)
```

<그림 4-21> 파일 생성자

파라미터 filename은 저장될 파일의 이름이고, 파라미터 mode에는 READ_ONLY, WRITE, WRITE_TRUNC, READ_WRITE가 올수 있다. 그러나 이 외의 값들이 들어오면 예외상황이 발생한다.

세 번째 파라미터 플래그는 <그림 4-22>가 온다.

```
FileSystem.PRIVATE_ACCESS
FileSystem.SHARED_ACCESS
FileSystem.SYSTEM_ACCESS
```

<그림 4-22> 플래그

FileSystem.PRIVATE_ACCESS에는 응용 프로그램이 자신만이 접근할 수 있는 디렉터리 파일을 생성하고자할 때 사용된다. FileSystem.SHARED_ACCESS는 공유할 수 있는 디렉터리에 파일을 생성하고자 할 때 사용된다. 공유하고자 하는 디렉터리는 이미 프로그램이 설치되어 응용 프로그램 명세파일(Jlet Descriptor)의 명시대로 지정되며, 사용자가 임의로 파일 공유 디렉터리를 변경할 수 없다. FileSystem.SYSTEM_ACCESS는 시스템 응용 프로그램이 사

용하는 디렉터리에 파일을 생성하고자할 때 사용한다. 공유하고자 하는 디렉터리는 이미 프로그램이 설치될 때 응용 프로그램 명세파일 명시대로 지정되며 사용자가 임의로 공유 디렉터리를 변경할 수는 없다. <표 4-1>은 접근 권한을 나타낸다.

<표 4-1> 접근 권한

변수명	값(이진수)	값(이진수)
DIR_SYCServerREAD_REQ_MASK	1	0000 0001
DIR_SYCServerWRITE_REQ_MASK	2	0000 0010
DIR_SHARD_READ_REQ_MASK	4	0000 0100
DIR_SHARD_WRITE_REQ_MASK	8	0000 1000
NETWORK_ACCESCServerREQ_MASK	16	0001 0000
SERIAL_ACCESCServerREQ_MASK	32	0010 0000
SYSTEM1_ACCESCServerREQ_MASK	64	0100 0000
SYSTEM2_ACCESCServerREQ_MASK	428	1000 0000

파일 생성 시에 고려해야 될 사항으로는 구현 환경이 개인용 컴퓨터가 아니라 휴대폰이므로 파일의 크기가 무제한으로 커질 수 없다. 따라서 파일의 크기를 지속적으로 관리해야할 필요가 있다.

본 논문에서는 파일의 크기를 관리하기 위한 최근최소사용 알고리즘인 LRU를 이용한다. LRU는 운영체계의 페이지 교체 알고리즘 중 하나로서 최근에 가장 적게 사용되어진 데이터를 제거하기 위해 사용된다.

<그림 4-23> 상용어구 실행화면

<그림 4-23>은 상용어구의 기능이 구현된 모습이다. Lotto라는 방 제목을 가진 대화방에 현재 “임창묵”이란 별명을 가진 사용자와 “Tester”란 별명을 가진 사용자가 서로 대화중이다. “임창묵”이란 사용자가 상용어구 입력키인 “*”키를 입력하고 이어서 “ㅅ”이라는 문자를 입력하였다. “ㅅ”이라는 문자를 인식한 후 상용어구 기능 창에 입력되어 있는 문자 중 “ㅅ”으로 시작하는 문자열을 순서대로 보여준다. 정렬기능 없이 입력된 순서대로 문자가 나타나게 된다. 이 창에서 입력할 문자의 번호를 선택한 후 확인 버튼을 누르면 선택한 문자가 대화창에 입력되고 다시 커서가 보이게 된다.

4.4 실험결과 및 비교분석

본 논문에서 구현한 문자 전송 서비스에서 상용어구 기능의 효율성을 테스트하였다. <표 4-2>는 단순 입력방식과 구현된 알고리즘의 상용어구 입력방식을 비교한 것이다. 입력은 총 50회의 테스트를 거쳤으며 평균입력 문자의 수가 5,200자의 한글과 특수문자로 이루어져 있다.

<표 4-2> 단순 입력 방식과 상용어구 입력방식 비교

구분	단순 입력 방식	상용어구 입력 방식
문자 수	5,200 자	5,200 자
단어 수	1,250 단어	1,250 단어
입력타수	13,000 타	6,700 타
Hit rate	0%	68.4%
입력비율	100%	51.5%

단어의 수로 환산하면 약 1,250 단어가 되며 이 기준이 되는 문장은 소설, 신문, 교재 등에서 무작위로 발췌한 문장과 대화방에서 주고받은 문장들로 구성되어 있다. 편지 한통을 쓴다고 할 때 대략 A4용지 3장 정도면 단순 입력 방식을 기준으로 하여 입력 타수의 측정치는 평균 13,000타이며, 이는 상용어구 방식의 단축키와 해당 단어의 검색 및 적용 과정에서 누를 수 있는 키 입력 타수의 측정치는 평균 6,700타로 집계되었다.

Hit rate는 단순 입력 방식이 전혀 참조되지 않고 입력된 경우로 참조율 0%일 때와 비교하여 상용어구 입력방식은 이미 입력된 단어들과 문장 입력 과정에서 재생산된 새로운 단어의 등록 및 재사용을 고려한 통계 값이다. 이것은 이미 입력되어 있는 단어들로써 국어사전과 대화시 자주 등장하는 단어들로 구성된 단어 검색 테이블로 총 3400여개를 이용하였다. 만약, 이러한 단어의 수를 더 늘린다면 Hit rate는 더 증가할 것이다.

입력비율은 단순 입력 방식을 100%기준으로 할 때 상용어구 입력방식은 51.5%로 두 배의 효율을 기대할 수 있었다. 따라서 단순 입력 방식보다 상용어구 입력 방식을 이용했을 때 수치상으로 효율적임을 알 수 있지만, 실제 단축키 방식의 상용어구 입력은 습관적으로 누를 수 있는 “*”와 “#” 기호 및 숫자 버튼 방식으로 사용에 있

어서 가속도가 붙을 수 있다. 실험자의 상용어구 입력 방식에 대한 숙련도 또한 처리 효율을 높이는데 영향이 있음을 알 수 있다. 이렇게 고려될 수 있는 요소들인 단어 검색 테이블의 입력 단어의 수, 사용자의 입력 방식에 대한 숙련도에 따라 단순 입력 방식에 비해 처리 가용도가 높아짐을 확인하였다.

제 5 장 결 론

오늘날 무선 인터넷의 급속한 발달과 함께 등장한 휴대폰은 급격한 발전을 이루고 있다. 과거에 예상하지 못했던 기능들이 추가됨으로서 단지 통화만을 위한 기기가 아니라 많은 기능들을 포함한 복합 단말기로 진화해 나가고 있다. 문자 전송에서 사진 촬영과 동영상 촬영 기능까지 갖추어 나가고 있다. 이렇게 발전해 나가는 휴대폰은 사용자들의 편의를 위해 더 많은 발전을 해 나갈 것이다.

본 논문에서는 현재 국내의 모바일 플랫폼의 사용 현황과 그로 인하여 발생하는 문제점과 그 문제점을 해결하기 위해 제안된 WIPI의 특징과 규격에 대하여 살펴보았다. 그리고 휴대폰 사용자들 간에 채팅을 할 수 있는 문자 전송 서비스에서 상용어구 기능의 적용에 대한 설계 및 구현을 하였다.

WIPI 플랫폼 기반에서 구현된 상용어구 기능은 다양한 단말기들 간의 표준으로 제조업체나 기종에 상관없이 WIPI를 지원하는 단말기기라면 실행될 수 있으므로 제조업체마다 별도의 프로그램을 각각 개발해야 하는 단점을 극복할 수 있었고, 단순 입력 방식의 경우 휴대폰 문자 메시지에 익숙한 젊은 세대들보다 문자 입력이 느린 장애인과 노령 사용자들에게 상용어구 입력 기능이 도움이 될 것으로 기대된다. 상용어구 기능이 갈수록 대중화되고 있는 정보통신의 물결에 주변인들을 참여시키는데 기여할 수 있을 것이다.

휴대폰 단말기들의 표준화를 위한 표준 플랫폼 기반인 WIPI의 우수성과 이를 기반으로 하는 다양한 응용과 실험들이 이루어질 때 정보통신 강국으로서 자리매김할 수 있을 것이다.

참고문헌

[1] 배석희, “모바일 표준 플랫폼 규격”, TTA 저널 82호, pp.59-66, 2002.

[2] 배석희, “모바일 플랫폼 표준화 동향 및 향후 발전방향”, TTA 저널 82호, pp.20-30, 2002.

[3] 김충만, 차세대 무선 인터넷 서비스, 전자신문사, 2003.

[4] 배석희, 한상홍, 전영준, 클릭하세요 위피, 대림, 2004.

[5] 박수원, 안은석, 이경철, 위피 모바일 프로그래밍, 한빛미디어, 2003.

[6] 자바누리, http://www.javanuri.net

[7] KTF Mobile Application Center, http://wipidev.magicn .com

[8] 한국 무선 인터넷 표준화 포럼(KWISF), 모바일 표준 플랫폼 규격 V1.2, V2.0, http://129.254.10.56/index.html, 2004

[9] 모바일랩 정보교육원, http://www.itkorea.or.kr

[10] 위피 개발자 커뮤니티, http://developer.wipi.or.kr

[11] 아로마소프트, http://www.aromasoft.com

[12] 모바일자바, http://www.mobilejava.co.kr

[13] 한국전자통신연구원, http://www.etri.re.kr

[14] 인트로모바일, http://www.intromobile.com

[15] K.mobile, http://www.kmobile.co.kr/index.asp

[16] 위피 Q&A 사이트, http://wipi.wisegram.com

[17] WIPI, http://wipi.or.kr

[18] 정의현, 김성진, 이기화, 조동찬, 클릭하세요 자바 2, 대림, 2003.

감사의 글

지난 2년 동안 부족한 저를 배려하고 지도해 주신 임재홍 교수님께 진심으로 감사드리며, 교수님의 지도와 격려에 더욱 열심히 했어야 했다는 아쉬움이 남아 죄송할 따름입니다. 또한 바쁘신 일정에도 논문을 지도해 주시고 심사해 주신 양규식 교수님, 박동국 교수님께도 진심으로 감사드립니다. 그리고 학교생활을 지도해 주신 학과 교수님께도 감사드립니다.

아울러, 논문을 완성하기까지 아낌없이 지원을 해 주신 신송아 선배님, 유선영 선배님, 김창수 선배님, 정성훈 선배님, 모수종 선배님께 감사드립니다. 또한 생활을 같이 하며 지내던 조원희 군, 최재석 군과 저에게 격려해 주었던 DCN_LAB 여러 선배님들에게도 고마움을 전합니다. 그리고 많은 도움을 주었던 박시형, 허 민, 최홍석 군에게도 감사의 인사를 전합니다.

항상 곁에서 도움을 준 친구들인 김이룩 군, 한지인 양에게 고마움을 전하며, 학과 동기들에게도 감사한 마음을 전합니다.

어려운 여건 속에서도 자식 뒷바라지를 하시며 격려와 용기를 주신 부모님께 감사드리며, 옆에서 큰 도움이 된 동생에게도 고마운 마음을 전합니다.

많은 분들의 도움으로 이루어진 논문인 만큼 더욱 열심히 살라는 가르침으로 알고 마음에 새기고 항상 노력하겠습니다.

본 연구는 산업자원부의 지역혁신 인력양성 사업의 연구결과로 수행되었음