텔레매틱스 단말기용 서비스 플랫폼 설계 및 구현☆

A Design and Implementation of Service Platform for Telematics Terminals

김 기 영* 김 동 균** 이 상 정***
Ki-Young Kim Dong-Kyun Kim Sang-Jeong Lee

요 약

본 논문에서는 자동차 관리 및 진단, 차량 편의장치 제어 등의 보다 향상된 텔레매틱스 서비스 제공을 위해 단말기용 서비스 플랫폼을 설계 구현한다. 서비스 플랫폼은 텔레매틱스용 소프트웨어를 개발하는 개발자에게 하부 자동차 내부 네트워크와 각종 하드웨어 장치, 무선 네트워크 장치에 독립적인 개발방법을 제공한다. 이를 위해 각 장치별로 제공되는 서비스를 분류하고 정형화한 텔레매틱스 API(Application Programming Interface)를 설계하였다. 제안된 서비스 플랫폼은 무선 네트워크 장치들과 단말기 간의 데이터 송수신을 위한 무선 접속부분과 자동차 내부 네트워크 장치들의 서비스를 위한 CAN 네트워크 부분, 위치기반 서비스를 위해 필요한 GPS 데이터 수신부분, 그리고 이기종의 네트워크 간 통신 시 발생하는 상이한 데이터 포맷의 상호 호환성을 위한 메시지 파서 부분으로 구성된다.

Abstract

In this paper, a telematics service platform, which supports the management and diagnosis of automobile and controls the automobile's convenience equipment, is designed and implemented. Using the service platform, telematics software can be developed independent of the lower part of automobiles's network devices, each hardware devices, and wireless network devices. The platform classifies each device's service and provides it with the fixed form of API(Application Programming Interface). This API consists of the wireless connection for a mobile device, the CAN network for controlling an automobile, the receiving part of GPS data for location base service, and the message parser for different data formats of different network.

☞ Keyword : Telematics, Telematics Service Platform, Bluetooth, CAN, GPS, PDA

1. 서 론

최근 정보통신과 자동차 기술이 결합하여 네비게이션, 위치추적, 인터넷 접속, 원격 차량진단, 사고감지, 교통정보 등을 제공하는 텔레매틱스 서비스가 크게 주목을 받고 있다. 텔레매틱스를 통해 자동차를 사무실과 가정에 이어 제 3의 인터넷 공간(Connected Car)으로 재구성함으로써 가정과 사무실에서 이용하는 서비스를 차량에서도 단절 없이 제공할 수 있다. 따라서, 자동차 안에서 보내는 시간을 보다 가치 있는 시간으로 활용하고자 하는 욕구를 기반으로 자동차와 정보통신 관련 산업을 중심으로 새로운 개념의 부가가치 서비스를 창출할 것으로 기대된다[1,2]. 텔레매틱스는 이제까지 서로 다른 산업이었던 자동차 산업과 정보통신 산업을 융합함으로써, 자동차 산업에는 경쟁력 강화를 통한 새로운 마케팅 기회와 수익을 제공하며, 이동통신 사업자에게는 새로운 사업 기회와 발전 가능성을 제공할 수 있을 것이다[3].

하지만 지금까지 국내의 텔레매틱스 서비스는 주로 교통정보 및 네비게이션 서비스 중심의 한

* 정 회 원 : (주)센트로닉스 방송통신SOC연구소 연구원
k71077@sch.ac.kr
** 정 회 원 : 순천향대학교 정보기술공학부 박사과정
kdk70@sch.ac.kr
*** 정 회 원 : 순천향대학교 정보기술공학부 교수
sjlee@sch.ac.kr
[2005/11/02 투고 - 2005/11/18 심사 - 2006/02/08 심사완료]
☆ 본 연구보고서는 정보통신부 정보통신연구진흥원에서 지원하고 있는 정보통신기초연구지원사업 No. B1220-0501-0145의 연구결과입니다.

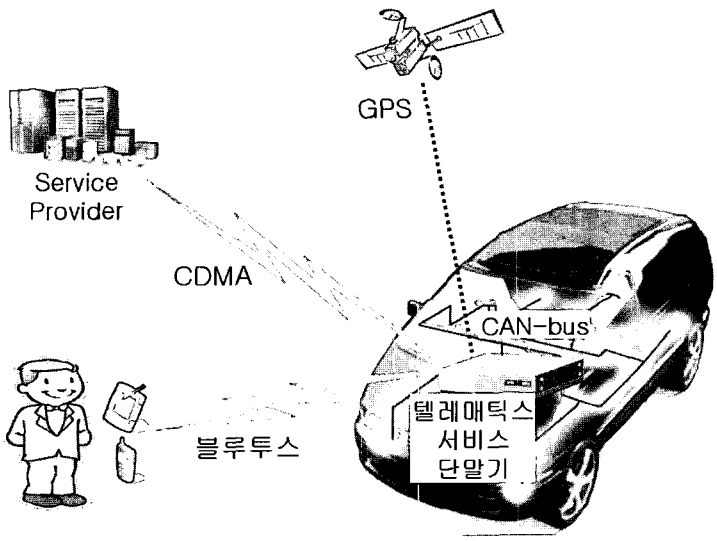
정된 서비스만이 제공되고 있다. 그러나 운전자에게 보다 향상된 텔레매틱스 서비스 제공을 위해서는 자동차 관리 및 진단, 차량 편의장치 제어 등의 기능이 추가되어야 한다. 이를 위해서는 자동차 내부의 네트워크 정보와 PDA(Personal Digital Assistant)나 핸드폰과 같은 운전자 개인의 모바일 단말기 사이에 원활한 데이터 교환을 위한 서비스 플랫폼이 요구된다.

본 논문에서는 자동차 관리 및 진단, 차량 편의장치 제어 등의 보다 향상된 텔레매틱스 서비스 제공을 위해 텔레매틱스 단말기용 서비스 플랫폼을 설계 구현한다. 제안된 서비스 플랫폼은 자동차 내의 다양한 네트워크 자원과 사용자의 PDA 및 외부 서비스 망에 접속하기 위한 텔레매틱스 단말기의 액세스 게이트웨이 상에서 응용 서비스 구현을 위한 소프트웨어 플랫폼이다. 서비스 플랫폼은 각 장치별로 서비스를 분류하고 정형화한 텔레매틱스 API(Application Programming Interface)를 제공하여 텔레매틱스용 소프트웨어 개발자가 하부 자동차 내부 네트워크와 각종 하드웨어 장치, 무선 네트워크 장치에 독립적인 응용 서비스를 개발 가능하게 한다. 또한 제안된 서비스 플랫폼의 적용을 위해 텔레매틱스 서비스 단말기를 개발하고 서비스 플랫폼 상에서 응용 서비스 시나리오를 구현하여 개발된 텔레매틱스 서비스 플랫폼의 타당성을 보인다.

2. 텔레매틱스 서비스 단말기

그림 1은 텔레매틱스 서비스 단말기의 연결 구성도이다. CAN(Controller Area Network) 버스를 통하여 획득된 자동차 내부의 정보가 블루투스 등의 근거리 무선망을 통해 개인의 정보단말이나 핫스폿에 전달되고, CDMA(Code Division Multiple Access)망을 통하여 서비스 제공자에게 전달된다. 즉, 자동차의 정비 및 고장과 관련된 자동차진단, 주행거리 및 운전습관에 따른 소모성 부품 교체 주기 통보와 같은 유지관리 서비스를 위한 자동차 내의 각종 디바이스의 정보 등이 전달된다. 또한 서비스 제공자로부터의 위치기반 서비스나 자동차 도어, 시트, 윈도우 등 편의장치 제어 데이터가 CAN 버스를 통하여 전달된다.

현재 자동차 내에 존재하는 윈도우, 라이트,



〈그림 1〉 텔레매틱스 서비스 단말기 연결 구성도

와이퍼 등 많은 종류의 사용자 편의장치들과 자동차 운행의 핵심을 담당하는 엔진 등의 각종 하드웨어 장치들은 기계적으로 또는 소프트웨어적으로 제어가 가능하다. 이러한 많은 하드웨어 장치들은 개발비용 감소 및 효율적인 관리를 위해 하나의 CAN 버스에 연결되고 있다[4-7]. 자동차의 각종 장치들은 텔레매틱스 서비스 단말기에 장착된 CAN 컨트롤러와 데이터를 송수신함으로써 좀더 지능화된 연동서비스가 가능하다.

그림 2는 설계 구현된 텔레매틱스 서비스 단말기의 아키텍처를 보여준다.

〈그림 2〉 텔레매틱스 서비스 단말기 아키텍처

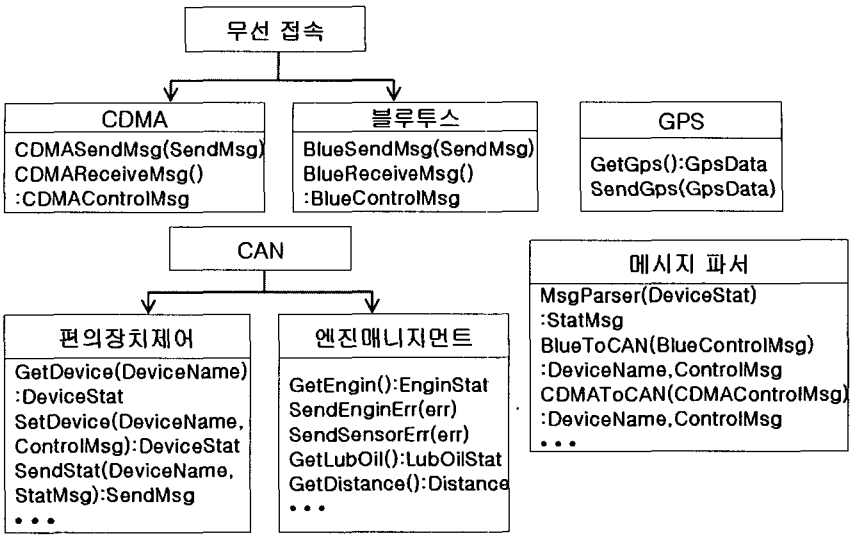
텔레매틱스 서비스 단말기에는 자동차의 각종 장치들과의 통신을 위해 PCI(Peripheral Component Interconnect) 방식의 CAN 컨트롤러를 장착하고 CAN 액세스 스택과 디바이스 드라이버가 구현되었다. 또한 자동차 상태 진단 및 유지관리 서비스, 편의장치 제어 및 모니터링 서비스를 위해서 사용자의 모바일 단말 및 근거리 핫스폿에 연결하기 위하여 블루투스 장치를 연결하고 블루투스 스택이 내장된다[8,9]. 도난방지 또는 네비게이션과 같은 위치기반 서비스를 위해 GPS(Global Positioning System) 데이터를 필요로 한다. GPS 데이터를 수신하기 위한 USB(Universal Serial Bus)방식의 GPS수신기를 장착하였다. 그리고 CDMA망을 통하여 텔레매틱스 서비스 제공자에게 연결하기 위하여 CDMA 모듈을 연결하고 CDMA액세스 스택이 내장된다. CDMA 액세스 스택에 있는 AT 명령세트를 이용하여 서비스 제공자의 고유번호를 통해 메시지를 송수신하게 된다.

3. 소프트웨어 플랫폼

텔레매틱스용 단말기에서는 여러 가지 역할을 하는 다양한 각종 자동차 네트워크 및 무선 네트워크 장치들과 장비 등으로 인해 일관된 방법의 응용 소프트웨어 개발이 어렵다. 따라서 하부 자동차 네트워크와 하드웨어 장치, 무선 네트워크에 독립적인 소프트웨어 플랫폼을 제공하여 일관된 방법으로 응용 프로그램을 개발하고 유지보수가 용이한 소프트웨어 플랫폼을 제공해야 한다[10,11]. 본 논문에서는 이를 위해 각 무선 접속기기 별로 제공되는 서비스를 분류하고 정형화한 텔레매틱스 API(Application Programming Interface)를 설계하였다. 자동차 내부 장치를 4가지 형태로 분류하고 고유한 식별자 값의 범위를 설정하여 표준 CAN 메시지 포맷에 따라 제어메시지를 설계하였다. 자동차 내부에는 다양한 네트워크 장치들이 존재 한다. 텔레매틱스 서비스를 원활히 수행하기 위해서는 메시지 포맷이 다른 이기종의 네트워크 간 통신이 빈번하게 발생한다. 따라서 서로 다른 메시지의 호환성을 위해 메시지 파서를 설계하였다.

3.1. 텔레매틱스 API 설계

그림 3은 소프트웨어 플랫폼을 위하여 정형화한 일부 API 함수를 보여주는 그림이다. 본 논문의 소프트웨어 플랫폼은 크게 네 가지 부분으로 구성된다. 무선 네트워크 장치들과 텔레매틱스 단말기의 데이터 송수신을 위한 무선접속 부분과 자동차의 각종 네트워크 장치들의 서비스를 위한 CAN 네트워크 부분, 위치기반 서비스를 위해 필요한 GPS 데이터를 수신하는 부분,



〈그림 3〉 텔레매틱스 API 함수

그리고 이 기종의 네트워크 간 통신 시 발생하는 상이한 데이터 포맷의 상호 호환성을 위한 메시지 파서 부분으로 구성하였다. 이러한 구분은 텔레매틱스용 소프트웨어를 개발하는 개발자에게 하부 자동차 내부 네트워크와 각종 하드웨어 장치, 무선 네트워크에 독립적인 개발방법을 제공해 준다[12].

블루투스용 API 함수들은 사용자의 모바일 단말을 통한 도어, 윈도우 등과 같은 자동차의 각종 편의장치 제어와 엔진 매니지먼트의 모니터링 서비스, 근거리 핫스폿 지역에서 차량진단 및 유지관리 서비스 등을 위해 사용된다. CDMA용 API 함수들은 근거리 핫스폿 지역을 벗어난 곳에서의 에어백 시스템의 사용과 같은 중대한 사고 발생 시 사고처리 등에 이용되며, 도난방지와 같은 서비스를 위해 GPS 데이터를 연동하여 서비스 제공자에게 자동으로 정보를 전송하는데 사용된다. 소프트웨어 플랫폼은 급변하는 기술 및 사용자의 요구 사항들로 인해 추가 및 수정이 용이해야 한다. 따라서 최소한의 가장 핵심적인 기능을 할 수 있도록 설계하고 구현한다.

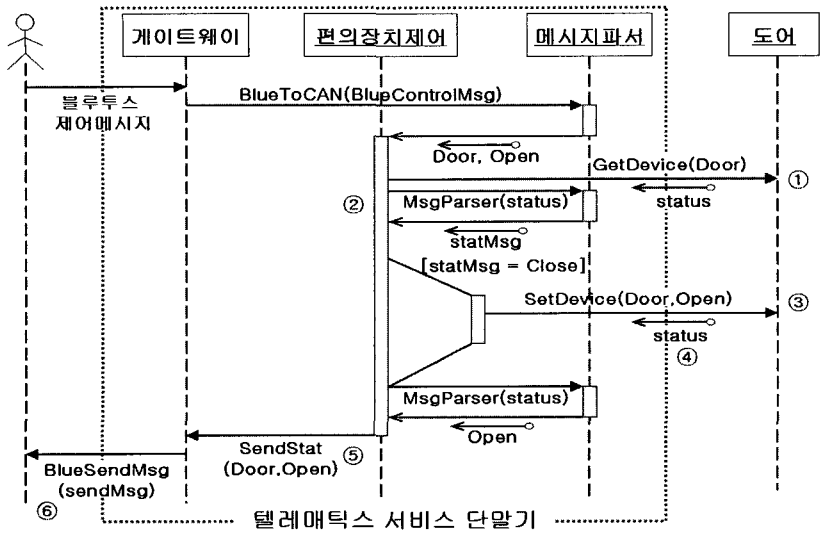
다음 그림 4는 API에 대한 시퀀스 다이어그램이다. 여러 가지 편의장치 제어 서비스 중에서 자동차 도어 Open 서비스에 대한 시퀀스 다이어그램의 예를 보여주고 있다. 사용자는 휴대하고 있는 모바일 단말인 PDA를 이용하여 텔레매틱스 서비스 단말기(게이트웨이)에 블루투스 접속을 한다. 이후 도어 Open 제어 메시지를 송신하게 된다. 텔레매틱스 단말기는 이 메시지를 메시지 파서에 넘겨주게 되고 메시지 파서에 의해 메시지 내용을 해석하게 된다. 편의장치 제어 서비스에서 도어 Open이라는 내용을 전달받게 된다.

편의장치 제어 서비스는 CAN 버스를 통해 이 메시지를 도어장치에 다음과 같은 순서로 송신하게 된다.

① GetDevice()함수를 이용하여 도어의 현재 상태를 얻어온다.

② 메시지 파서를 통해 도어의 현재 상태를 해석한다.

③ 현재 상태가 Close인 경우 SetDevice()함수를 이용하여 도어를 Open하게 된다. 이때 현재 상태가 Open인 경우 현재 상태를 바로 반환한다.



〈그림 4〉 API 시퀀스 다이어그램

④ Open된 도어는 자신의 상태를 편의장치 제어 서비스에 반환

⑤ SendStat()함수를 이용하여 블루투스용 메시지 생성 후 게이트웨이에 전달

⑥ BlueSendMsg()함수를 이용하여 사용자의 모바일 단말인 PDA에 블루투스 통신을 통하여 상태를 반환

3.2. CAN 제어 메시지

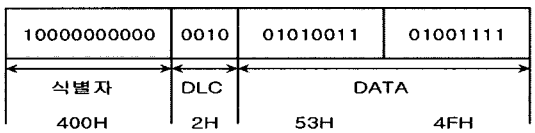
표 1은 CAN 제어 메시지의 11비트로 구성되는 식별자 값을 설정하기 위한 구분과 범위를 나타낸다.

〈표 1〉 11비트 식별자 구분과 범위

구 분	범 위
실시간 제어 애플리케이션 (안정성 제어, 엔진관리, 변속 등)	100H - 2FFH
일반적인 기능 (파워 윈도우, 계기 등)	300H - 3FFH
편의기능 및 고급기능 (전자부트 작동기, 썬루프 등)	400H - 5FFH
멀티미디어 서비스 (인터넷, DTV 등)	600H - 7FFH

표 1과 같이 식별자의 값이 100H~2FFH까지의 범위를 갖는 장치는 파워 트레인, 안정성 제어(ABS, 견인제어, 액티브 서스펜션), 엔진관리, 변속 같은 실시간 제어 애플리케이션에 사용되는 장치들이 사용한다. 식별자 값이 300H~3FFH까지의 범위를 갖는 장치는 파워 윈도우, 좌석 조절장치, 계기 같은 일반적인 장치들이 사용한다. 식별자 값이 400H~5FFH까지의 범위를 갖는 장치는 전자 부트 작동기, 전동 거울 조정장치, 비 탐지, 썬루프, 기상관리 등과 같은 편의 기능 또는 고급 기능들을 하는 장치들이 사용한다. 식별자 값이 600H~7FFH까지의 범위를 갖는 장치는 인터넷, 음성 및 동영상 등과 같은 멀티미디어 서비스를 제공해 주는 장치들이 사용한다.

다음 그림 5는 CAN 메시지 형식에 따른 도어 Open 제어 메시지의 예를 보여준다.



〈그림 5〉 CAN 제어 메시지 예

도어에 대한 유일한 식별자 값은 16진수 400H로 설정한 값이고, 제어 데이터 길이(DLC : Data Length Code)는 2바이트의 데이터를 나타내는 16진수 2H를 나타낸다. 실제 제어 데이터는 2바이트로 나타낼 수 있는데 첫 번째 바이트는 Set을 의미하는 'S'의 아스키 코드값 53H, 두 번째 바이트는 Open을 의미하는 'O'의 아스키 코드값 4FH로 나타낸다.

텔레매틱스 단말기의 게이트웨이는 사용자의 PDA로부터 블루투스 통신을 통해 도어 Open에 대한 메시지를 수신한 후 메시지 파서에 의해 도어 Open에 대한 메시지임을 해석한 후 그림 5와 같은 메시지를 생성한다. 이후 CAN 버스에 연결된 모든 장치들은 이 메시지를 수신한 후 식별자 값을 확인하여 자신의 데이터가 아니면 메시지를 무시하게 된다. 도어는 이 값을 수신한 후 메시지를 해석해서 자신의 식별자 값인 400H와 비교를 한 후 자신의 데이터임을 확인한 후 도어를 Open상태로 변경하게 된다.

3.3. 제어 메시지 파서

텔레매틱스 단말기에는 여러 가지 역할을 하는 다양한 각종 자동차 장비 및 자동차 하부 네트워크와 사용자 및 외부 네트워크와의 연결을 위한 무선 네트워크 장치들이 존재한다. 따라서 이러한 하부 자동차 네트워크와 하드웨어 장치, 각종 무선 네트워크 간 통신이 빈번하게 발생하게 된다.

본 논문에서는 이 기종의 네트워크 간 통신 시 발생하는 가장 큰 문제점인 메시지 호환성을 위해 메시지 변환을 담당하는 메시지 파서를 설계하였다. 그림 6은 메시지 변환을 위한 주요 자료구조 정의부분을 나타내고 있다.

사용자로부터 텔레매틱스 단말기로 송신되는 제어메시지 정보는 일반 텍스트 형태의 메시지와 CDMA SMS(Short Message Service) 메시지 형태의 두 가지로 구분되어 전송되게 된다. 사용자 및 근거리 핫스폿 지역에서 블루투스 통

```
#ifndef UINT8
#define UINT8 unsigned char
#endif
#ifndef UINT16
#define UINT16 unsigned short
#endif
#ifndef UINT32
#define UINT32 unsigned long
#endif
.....

//==============================
//      Vehicle Device ID
//==============================
#define PCU         0x100
#define ECU         0x101
#define ES          0x102
#define AIRV        0x103
#define AIRFS       0x104
#define AIRP        0x105
#define ECIM        0x106
#define TPS         0x107
#define CEFP        0x108
#define PCSV        0x109
#define TCU         0x140
#define DOOR        0x400
#define WINDDOW     0x300
#define ROOF        0x410
#define ECM         0x420
.....
```

```
//==============================
//      Message information
//==============================
#define SET         0x53
#define GET         0x47
#define OPEN        0x4f
#define CLOSE       0x43
#define ERR         0x45
.....

typedef struct
{
  UINT32 id;                /* message identifier */
  UINT8 dlc;                /* Data Length Code (0-8) */
  UINT8 mdf;                /* message frame format */
  UINT8 a_data[8];          /* CAN data */
  UINT32 time_stamp;        /* receive time stamp */
} CANMsg;

typedef struct
{
  char PhoneNumber[11];
  char Msg[80];
} CDMAMsg;
.....
```

〈그림 6〉 메시지 자료구조 정의

신을 통해 전송되는 제어 메시지는 일반 텍스트 형태로 전송되고 원거리의 사용자 및 TSP (Telematics Service Provider)로부터 전송되는 메시지는 CDMA SMS 메시지 형태로 전송된다. 전송된 제어메시지는 자동차 하부 네트워크인 CAN을 이용하여 각종 장치들을 제어하고 모니터링 해야 하기 때문에 CAN 메시지 형태로의 변환과정을 거쳐 CAN 버스를 통해 메시지를 실제 장치에 전송하게 된다. 실제 장치에서는 전송 받은 CAN 메시지를 해석하여 메시지에 해당하는 동작을 하게 되고 동작 후에 상황을 다시 재전송하게 된다.

그림 7은 사용자로부터 전달 받은 메시지를 CAN 메시지 형태로 변환 후 차량창치를 제어하고 제어 후의 상태인 CAN 메시지를 사용자에게 다시 반환 해주는 과정을 나타내는 메시지 흐름도이다.

제어 메시지의 송신에서부터 제어 후의 상태를 모니터링 하기까지의 과정은 크게 7가지 부분으로 나누어 볼 수 있다.

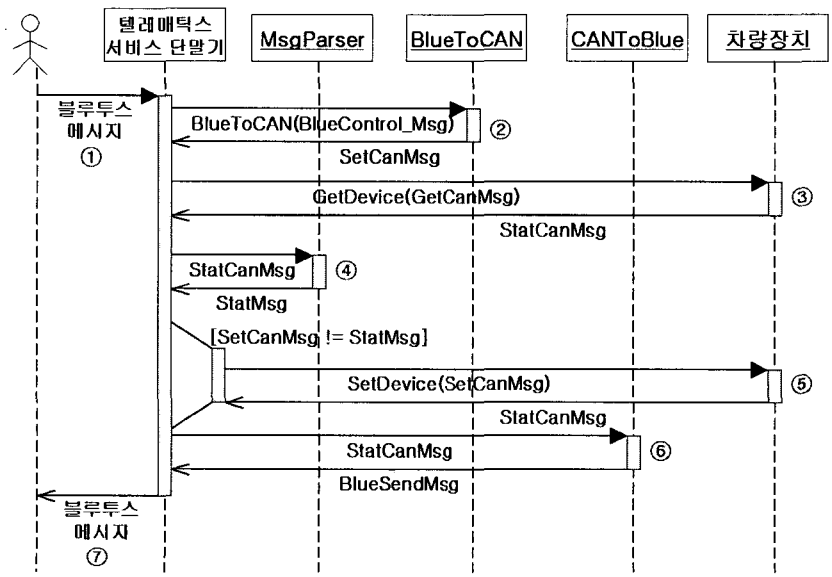
① 텔레매틱스 단말기 시스템에서는 사용자로부터 일반 텍스트 형태의 제어메시지인 BlueControl_Msg를 수신한다.

② 수신된 BlueControl_Msg는 BlueToCAN() 함수에 전달되어 토큰 분류를 통해 제어하고자 하는 장치명과 제어 메시지를 분류하여 해당되는 장치를 제어하기 위한 CAN 메시지 형태인 SetCanMsg를 생성한다. 생성된 SetCanMsg는 텔레매틱스 시스템에 반환되게 된다.

③ 제어하고자 하는 장치의 현재 상태를 알아보기 위해 SetCanMsg의 식별자 값을 사용하여 CAN 메시지 형태인 GetCanMsg를 생성, 차량장치에 현재의 상태를 요청한다. 차량장치는 현재 상태를 CAN 메시지 형태인 StatCanMsg를 텔레매틱스 시스템에 반환하게 된다.

④ 위 과정에서 반환된 StatCanMsg를 MsgParsor() 함수에 전달하여 현재 상태 정보인 StatMsg를 추출한다. 추출된 StatMsg는 텔레매틱스 시스템에 반환한다.

⑤ 반환된 StatMsg와 ②번 과정에서 생성된 SetCanMsg의 제어메시지를 비교하여 현재



〈그림 7〉 메시지 흐름도

상태와 제어하기 위한 상태를 판단한다. 이때 제어하기 위한 상태와 현재상태가 같지 않다면 현재상태를 변경시켜줘야 한다는 것을 의미하기 때문에 SetCanMsg를 차량장치에 송신하여 장치를 제어하게 된다. 제어 후의 상태인 StatCanMsg를 다시 텔레매틱스 시스템에 반환하게 된다.

⑥ 반환된 StatCanMsg를 다시 사용자에게 알려주기 위해 CANToBlue() 함수에 전달된다. 또한 사용자에게 변환 후의 정보를 알려주기 위해서 일반 텍스트 형태의 메시지인 BlueSendMsg를 생성하여 텔레매틱스 시스템에 반환한다.

⑦ 텔레매틱스 단말기 시스템에서는 반환된 BlueSendMsg를 사용자에게 전달함으로써 제어과정을 완료하게 된다.

4. 응용 서비스 시나리오 적용

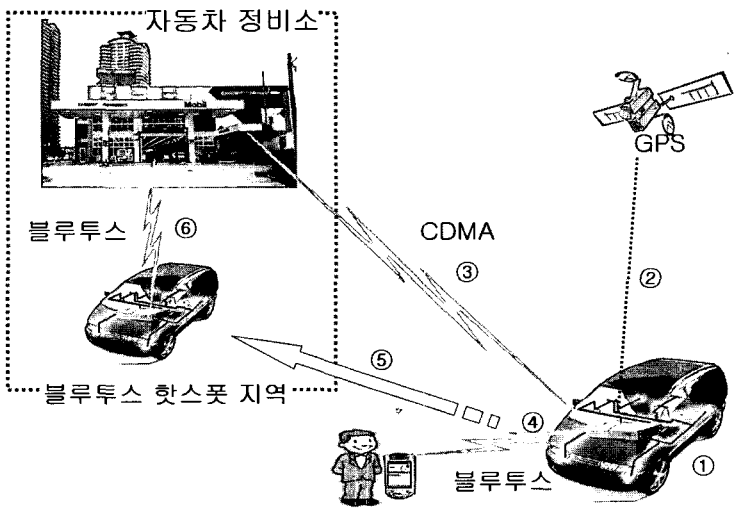
그림 8은 텔레매틱스 응용서비스 시나리오에 대한 그림이다. 사용자는 블루투스 통신이 가능한 모바일 단말기인 PDA를 가지고 있다. 사용자는 자동차의 자가 진단서비스인 소모성 부품 상태 및 차량의 이상부품 발생여부를 PDA로 모니터링 한 후 이상부품 발생과 소모성 부품의 교체시기를 전달 받게 된다.

서비스 시나리오는 크게 6단계로 구분해서 설명하면 다음과 같다.

① 사용자의 자동차 운행도중 자동차 부품의 이상을 감지한 후 자동차의 디바이스 장치에서는 CAN 버스를 통해 CAN 데이터 형식의 오류 메시지를 텔레매틱스 단말기에 전달한다.

② 텔레매틱스 단말기는 이상이 발생한 부품을 교체 서비스를 받아야 한다는 것을 사용자에게 알려주어야 한다. 현재 자동차의 위치를 파악하기 위해 GPS 수신기로부터 GPS 데이터를 얻어온다. 현재 자동차의 위치를 중심으로 가장 가까운 자동차 정비소를 탐색한다.

③ 탐색된 자동차 정비소에 CDMA SMS를 이용하여 현재 이상이 발생한 부품 정보를 보내 재고 유무를 요청한다. 자동차 정비소에서는 재고 유무를 파악한 후 수신된 번호를 통해 재고 유무를 임베디드 텔레매틱스 시스템에 재전송 하게 된다. 재고 여부에 따라 자



〈그림 8〉 응용서비스 시나리오

동차 정비소 방문 예약 메시지를 송신한다.

④ 텔레매틱스 단말기는 블루투스로 연결되어 있는 사용자의 PDA에 이상이 발생한 부품에 대한 정보와 현재 위치에서 가장 가깝고, 이 상부품 재고를 보유하고 있는 방문 예약한 자동차 정비소의 정보를 송신하게 된다.

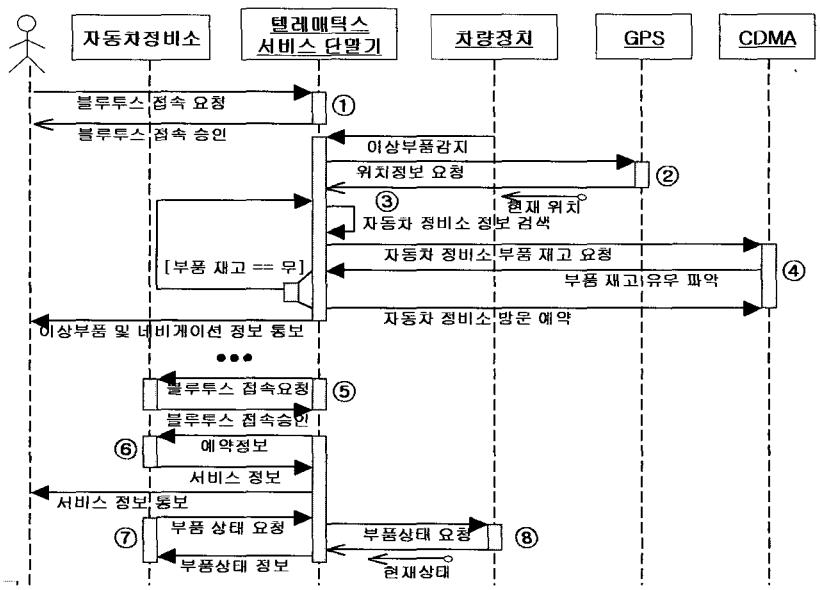
⑤ 사용자는 가장 가까운 자동차 정비소에 대한 정보를 네비게이션 서비스를 이용하여 최적의 경로로 이동하게 된다.

⑥ 자동차 정비소에는 블루투스 통신을 이용하여 근거리 핫스폿을 형성하고 있다. 따라서 사용자의 자동차가 자동차 정비소의 근거리 핫스폿 지역에 진입하는 순간 자동차 내의 텔레매틱스 단말기는 자동차 정비소의 블루투스 AP(access point)와 블루투스 접속을 요청한 후 블루투스 연결설정을 완료하게 된다. 블루투스 연결설정이 이루어진 후에는 텔레매틱스 단말기에서 방문 예약 정보 메시지를 자동차 정비소에 송신하고, 자동차 정비소에서는 접수번호와 담당자, 정비 라인 번호를 전송한다.

그림 9는 응용 서비스 시나리오에 대한 서비스 흐름도이다.

① 과정에서 사용자와 텔레매틱스 단말기 간의 블루투스 접속을 수행한다. 다음 그림 10은 위 서비스 흐름도에서 ②부터 ④까지의 과정을 함수 호출과정을 통해 설명하고 있다.

②의 과정에서 자동차의 부품인 에어플로우 센서의 이상을 감지한 후 CAN 네트워크를 통해 임베디드 시스템에 error 상태 메시지를 전송하게 된다. 텔레매틱스 단말기에서는 MsgParser 함수를 이용하여 현재 상태를 판단하게 되고, 현재상태가 ERR 일 경우 GPS수신기로부터 현재 자동차의 위치 정보를 GetGps 함수를 이용하여 수신하게 된다. ③의 과정에서 수신된 자동차의 위치정보를 네비게이션 서비스 데이터베이스에서 가까운 자동차 정비소의 상호를 Search-Service 함수를 이용하여 얻어온다. 또 이동중인 자동차이기 때문에 CDMA 통신을 이용해야함으로 정비소의 상호를 바탕으로 지역 전화번호 데이터베이스에서 CDMA번호를 SearchPhone함수를 통해 얻어온다. ④의 과정에서 현재 error의



〈그림 9〉 서비스 흐름도

```
int telematic_service (int blue_fd, int can_fd, int cdma_fd) {
   .....
   FD_SET(blue_fd, &read_fds);      //bluetooth 통신
   FD_SET(can_fd, &read_fds);       //CAN 통신
   FD_SET(cdma_fd, &read_fds);      //CDMA 통신
   char phonenum[11]; char servicecenter[80]; char position[70];
   CDMAMsg cdmaSendMsg, cdmaReceiveMsg;
   char errStatMsg[80]; char reservation[80] = "1980차량 예약합니다.";
   if(select(maxfdpl, &read_fds, NULL, NULL, NULL) < 0) { ... }
   if (FD_ISSET(can_fd, &read_fds)) {
      BCI_ReceiveCanMsg (CANBoard, Controller, &receiveMsg, BCI_NO_WAIT);
      UINT8 status = MsgParser(receiveMsg);
      if (status == ERR) {
         while(1){
            position = GetGps();      ②
            servicecenter = SearchService(position);
            phonenum = SearchPhone(servicecenter);
            errStatMsg = CANToCDMA(receiveMsg);      ③
            cdmaSendMsg->PhoneNumber = &phonenum;
            cdmaSendMsg->Msg = &errStatMsg;
            CDMASendMsg(cdmaSendMsg);
            while (cdmaReceiveMsg->Msg != NULL)
            { cdmaReceiveMsg = CDMAReceiveMsg(); }
            if ( !strcmp(cdmaReceiveMsg->Msg, "OK")) break;
         }
         cdmaSendMsg->Msg = &reservation;
         CDMASendMsg(cdmaSendMsg);      ④
      } }
```

〈그림 10〉 정비소 예약과정까지의 호출과정

```
int telematic_service (int blue_fd, int can_fd, int cdma_fd) {
   .....
   FD_SET(blue_fd, &read_fds);      //bluetooth 통신
   FD_SET(can_fd, &read_fds);       //CAN 통신
   FD_SET(cdma_fd, &read_fds);      //CDMA 통신
   CANMsg BlueControlMsg, statMsg;
   char RetReservation[80] = "1980 차량입니다";
   char errStatMsg[80]; char reservation[80] = "1980차량 예약합니다.";
   char receiveBlueMsg[256]; char sendBlueMsg[256];
   if(select(maxfdpl, &read_fds, NULL, NULL, NULL) < 0) { ... }
   if (FD_ISSET(blue_fd, &read_fds))
   {
      receiveBlueMsg = BlueReceveMsg();
      if (strcmp(receiveBlueMsg, "welcom"))
      {
         BlueControlMsg = BlueToCAN(&receiveBlueMsg);
         if ( BlueControlMsg->id != 0)
         {
            statMsg = GetDevice(BlueControlMsg->id); ⑦
            sendBlueMsg = CANToBlue(statMsg);
            BlueSendMsg(sendBlueMsg)  ⑧
         }
         BlueSendMsg(&receiveBlueMsg);
      }
      else BlueSendMsg(&RetReservation);⑥
   }
}
```

〈그림 11〉 정비소 도착 후 호출과정

상태정보는 CANMsg 형식으로 되어 있기 때문에 CDMA메시지 형태로 변환하기 위해 CANToCDMA 함수를 통해 일반 텍스트 형태로 변환을 한다. 변환된 정보를 ③의 과정에서

얻은 자동차 정비소의 CDMA번호와 함께 CDMAMsg 형태의 메시지를 생성하고 CDMA-SendMsg 함수를 통해 자동차 정비소에 에어플로우 센서의 재고 여부를 문의하게 된다. 자동차 정비소에서는 이를 확인 후 바로 재고 여부를 재전송 해주게 되고 텔레매틱스 단말기에서는 이를 수신하여 재고가 있으면 현재 CDMA번호를 재사용해서 사용자 자동차의 번호를 이용하여 예약 메시지를 자동차 정비소에 CDMA-SendMsg 함수를 이용하여 송신하게 된다. 이와 동시에 사용자에게 이상부품 정보를 전송하고 네비게이션 시스템에 자동차 정비소의 위치를 보여주게 된다. 사용자는 네비게이션 서비스를 이용하여 빠르고 안전하게 자동차 정비소까지 이동을 하게 되고, 블루투스 근거리 핫스폿을 구성하고 있는 자동차 정비소에 진입하자마자 ⑤부터 ⑧까지의 과정이 이루어지게 된다. 다음 그림 11은 자동차 정비소에 도착했을 때의 과정이다.

⑤의 과정은 ①의 과정과 동일하게 동작을 한다. 즉 블루투스 근거리 핫스폿을 형성하고 있는 자동차 정비소에 도착하면 블루투스 장치를 인식하게 되고 연결설정을 자동적으로 실행하여 통신이 가능한 상태가 된다. 그 후 ⑥의 과정에서 1980 차량으로 예약했다는 메시지를 자동차 정비소에 BlueSendMsg 함수를 이용하여 송신하고, 자동차 정비소에서는 서비스를 담당할 담당자와 서비스 받을 장소인 0번 게이트의 정보를 보내주게 된다. 텔레매틱스 단말기에서는 담당자와 서비스 장소를 사용자에게 알려주게 되고 사용자는 서비스 장소로 이동을 한다. 서비스 장소 이동 후 ⑦의 과정에서 담당자는 블루투스를 이용하여 이상이 발생했던 에어플로우 센서의 상태를 다시 한번 요청하게 된다. ⑧의 과정에서 임베디드 시스템은 CAN통신을 이용하여 에어플로우 센서에 현재 상태를 요청하고 상태를 받는다. 받은 상태를 담당자에게 알려줌으로써 서비스를 받게 된다.

5. 구현 및 비교평가

5.1 구현 및 테스트

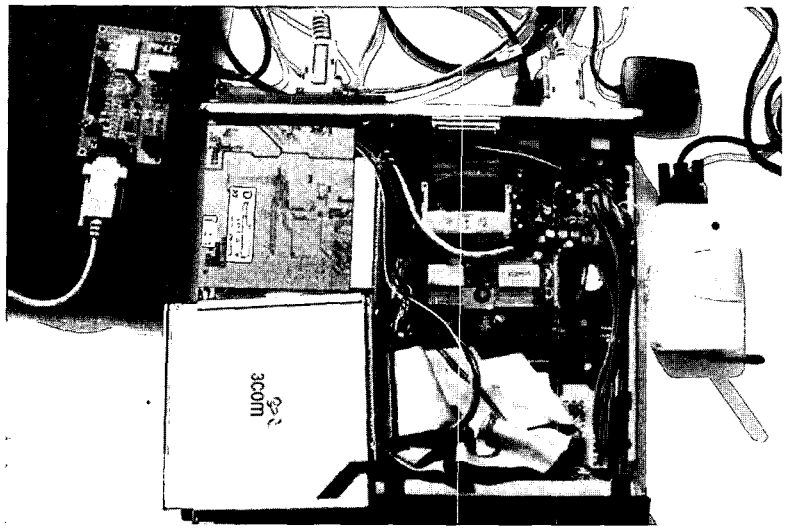
표 2는 본 논문에서 구현한 텔레매틱스 서비스 단말기의 구현환경이다.

〈표 2〉 텔레매틱스 서비스 단말기 구현환경

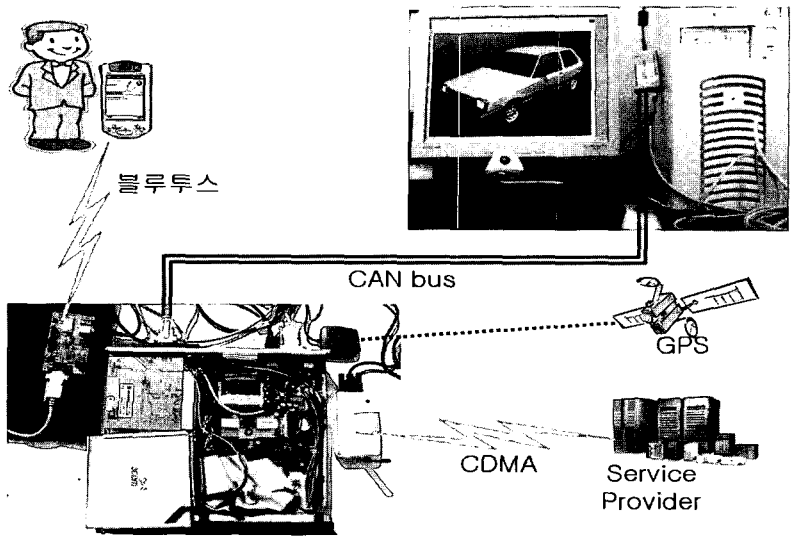
구 분	임베디드 텔레매틱스 시스템
프로세서	VIA C3/VIA Eden ESP 프로세서 1GHz
OS	레드햇 리눅스 9.0 (Linux Kernel 2.4.26)
블루투스 모듈	Seecode(주), bluebox, CSR
블루투스 스택	공개형 리눅스 공개스택 BlueZ
CDMA 모듈	KTF SMS 전용 샵가이더
GPS 수신기	인텔링스 i-3000U
CAN 모듈	IXXAT, PC-I 04/PCI

텔레매틱스 서비스 단말기의 환경은 VIA C3/VIA Eden ESP 1GHz 프로세서를 장착한 SBC(Single Board Computer)위에 Linux Kernel 2.4.26을 포팅하였다. 블루투스 통신을 위하여 CSR사의 블루투스 칩을 사용한 Seecode(주)의 Bluebox를 장착하고, 리눅스 블루투스 공개스택인 BlueZ를 사용하였다. GPS 수신을 위하여 SiRF star IIe칩을 사용한 인텔링스사의 i-3000U를, CDMA 모듈은 KTF SMS 전용모듈인 샵가이더를 장착하였다. CAN 통신은 필립스사의 SJA1000칩을 사용한 IXXAT사의 PC-I 04/PCI를 사용하였다. 다음 그림 12는 구현된 텔레매틱스 서비스 단말기로 블루투스 모듈과 CAN모듈, CDMA모듈, GPS모듈이 장착된 모습이다.

사용자는 텔레매틱스 서비스 단말기에 장착된 블루투스 장치의 통신가능 지역 내에 위치하는 동시에 사용자의 PDA에서 차량에 장착된 블루투스 장치를 감지하고 접속한다. 이후 사용자는 PDA를 이용하여 차량의 도어장치에 Door Open 메시지를 이용하여 차량의 도어를 열게 된다. 이때 Door Open 메시지는 블루투스 통신



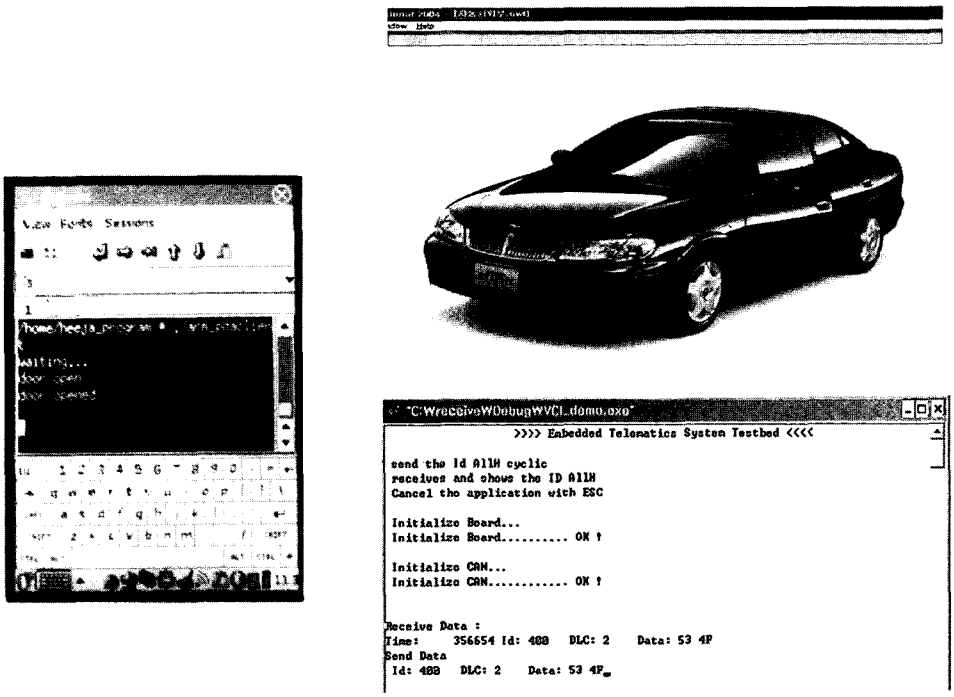
〈그림 12〉 구현된 텔레매틱스 서비스 단말기



〈그림 13〉 구현된 텔레매틱스 서비스 단말기 테스트 베드

을 이용하여 텔레매틱스 단말기의 서비스 게이트웨이에 전달이 되고, 메시지 파서에 의해 메시지를 해석하게 된다. 해석된 메시지는 편의장치 제어 서비스에 의해 실제 장치인 도어 장치에 전달이 되고 도어는 Open되게 된다. Open된 도어는 현재 상태를 다시 반환함으로써 사용자에게 차량의 도어가 Open되었다는 메시지를 전송하게 된다. 또한 윈도우나 썬루프와 같은 편의장치에 메시지를 보내 편의장치를 제어 및 모니터링하게 된다. 사용자는 다시 차량에서 내려 도보로 이동하면서 PDA를 사용하여 차량에 Door Close 메시지를 전송함으로써 차량의 도어장치를 Close하게 된다.

다음 그림 13은 구현된 텔레매틱스 서비스 단



〈그림 14〉 도어오픈 제어 시뮬레이션 결과

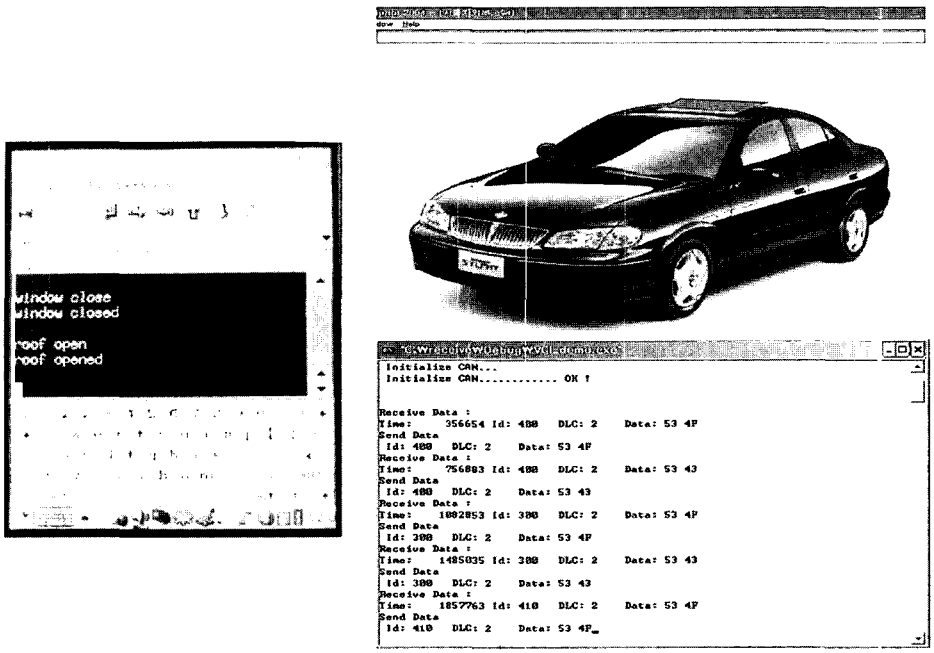
말기 테스트 베드의 환경을 보여준다. 텔레매틱스 서비스 단말기를 테스트하기 위해 팬티엄 4(2.4GHz) 프로세서의 컴퓨터상에서 자동차 시뮬레이션을 구현하여 테스트하였다. 자동차 시뮬레이션 컴퓨터에는 CAN 통신을 위해 필립스사의 SJA1000칩을 사용한 IXXAT사의 USB-to-CAN compact 모듈을 장착하여 텔레매틱스 서비스 단말기와의 CAN 통신을 수행하였다. 그리고 텔레매틱스 서비스 단말기와 사용자 모바일 단말인 PDA와 통신을 위해 블루투스 통신이 가능한 PDA상에서 편의장치 제어 서비스를 수행하였다.

구현된 텔레매틱스 서비스 단말기 테스트 베드를 검증하기 위해 자동차 시뮬레이션을 통해 제어동작을 확인하였고 편의장치 중에서 윈도우, 썬루프 등을 구현하여 테스트하였다.

그림 14는 도어 Open을 시뮬레이션한 그림으로 구현된 텔레매틱스 서비스 단말기에서 사용자의 PDA로부터 도어장치 오픈에 대한 메시지를 블루투스 통신으로 전송받는다. 전송받은 메시지는 CAN 제어메시지로 변환된 후 CAN버스를 통해 시뮬레이터에 전송된다. 전송된 메시지에 따라 도어장치를 오픈 한 상태를 나타낸다. 시뮬레이터에서는 도어장치를 오픈한 후의 CAN 메시지인 상태정보를 다시 CAN버스를 통해 텔레매틱스 서비스 단말에 전송한다. 전송받은 CAN 메시지는 사용자가 알아 볼 수 있는 블루투스 메시지로 변환된 후 사용자의 PDA에 블루투스 통신을 통해 전송함으로써 제어과정을 마친다. 그림 15는 썬루프 Open을 시뮬레이션한 그림으로 그림 14와 동일한 방법으로 동작을 수행한다.

5.2 비교평가

텔레매틱스 단말 플랫폼 미들웨어 대부분의



〈그림 15〉 루프오픈 제어 시뮬레이션 결과

규격을 운영체제에 독립적인 방법으로 JAVA 기반 위에서 구축하는 방식의 AMI-C(Automotive Multimedia Interface Collaboration), 다양한 서비스와 컨텐츠를 가정, 사무실 및 자동차에서 지원할 수 있도록 표준화된 Gateway에서 서비스 변경, 소프트웨어 변경 등을 지원하는 OSGi(Open Service Gateway Initiative), 하드웨어나 운영체제에 무관하게 응용서비스가 가능하도록 API를 제공함으로써 새로운 서비스를 자유롭게 추가할 수 있는 표준 플랫폼을 지향 하는 Java for Automotive, 고속 광통신 방식으로 25-150Mbps를 지원하는 MOST(Media Oriented Systems Transfer), 저속의 애플리케이션을 위한 LIN(Local Interconnect Network), 저전력의 고속 멀티미디어 서비스를 위한 UWB, 블루투스 등의 기술 표준화가 진행되고 있다[13].

본 논문에서 설계하고 구현한 시스템을 위와 같은 표준화 진행중인 기술들을 활용한 기존 연구방식과 비교하였다[8,10,13,14]. 구분을 쉽게하기 위해 다음과 같이 기존연구방식을 A,B,C,D로 구분하였다.

A : "A Bluetooth-CAN Gateway",

B : "Wireless Application Development in a Telematics Environment",

C : "차세대 텔레매틱스 서비스를 위한 통합 제어 플랫폼",

D : "텔레매틱스 단말 S/W 플랫폼 표준 아키텍처 설계"

본 논문에서 설계 구현된 텔레매틱스 단말기용 서비스 플랫폼을 표 3에서와 같이 기존 연구와 비교하였다. 기존 연구들과 비교하여 필요한 네트워크 부분을 모두 고려하였다. 텔레매틱스 응용 프로그램 개발자에게 개발 및 유지 보수가 용이하도록 각 네트워크 장치별로 사용가능한 API를 설계 하였다. 자동차의 각종 장치를 제어하기 위한 CAN 제어메시지를 설계하였고 이기종의 네트워크 통신 시 발생하는 상이한 데이터 포맷의 상호 호환성을 위해 메시지 파서를 설계하여 기존 연구와 차별화 하였다.

〈표 3〉 기존 연구들과의 특징 비교

구 분	기 존 연 구				본 연구
	A	B	C	D	
CAN 통신	적용	적용 안됨	적용 안됨	적용	적용, 구현
GPS 수신	적용 안됨	적용	적용	적용	적용 구현
무선통신	블루투스	WLAN (Wireless LAN)	무선 통신 언급	WLAN, UWB, 블루투스	블루투스, CDMA 적용 구현
텔레매틱스 API	적용 안됨	적용 안됨	적용 안됨	적용	설계, 구현
CAN 제어메시지	적용 안됨	적용 안됨	적용 안됨	적용 안됨	설계, 구현
제어메시지 파서	적용 안됨	적용 안됨	적용 안됨	적용 안됨	설계, 구현

6. 결론 및 향후 연구방향

본 논문에서는 하부 자동차 네트워크와 하드웨어 장치, 무선 네트워크에 독립적인 일관된 방법으로 응용 프로그램을 개발하고 유지보수가 용이하도록 하기 위해 각 무선 접속기기 별로 제공되는 서비스를 분류하고 정형화한 텔레매틱스 API를 설계하였다. 무선 네트워크 장치들과 텔레매틱스 서비스 단말기와의 데이터 송수신을 위한 무선 접속부분, 자동차의 각종 네트워크 장치들을 위한 CAN 네트워크 부분, 위치기반 서비스를 위해 GPS 부분, 이 기종의 네트워크 간 통신 시 발생하는 상이한 데이터 포맷의 상호 호환성을 위한 메시지 파서를 설계하였다. 또한 제안된 서비스 플랫폼을 사용한 응용 서비스 시나리오를 제시하고 이를 구현된 텔레매틱스 서비스 단말기 상에 적용하였다.

향후 설계 구현된 텔레매틱스 서비스 단말기에 DMB(Digital Multimedia Broadcasting) 수신 단말기를 연결하고, 실시간 교통여행 정보 수신을 위해 TPEG(Transport Protocol Experts Group) 메시지 송수신 처리 미들웨어를 구현할 계획이며, 편의장치 이외의 다른 장치들도 시뮬레이터를 구현하여 장치들간 연동 테스트를 진행할 예정이다.

참 고 문 헌

[1] 안선일, 장병준, 이윤덕 "텔레매틱스 동향 및 기술개발 방향", 한국 정보과학회 학회지, 제 23권 제2호 p.77-82, 2005년 2월

[2] Telematics Research Group Incorporated, http://www.telematicsresearch.com

[3] 김주완 "텔레매틱스 기술 및 서비스 동향", IITA 주간기술동향, 1144호, 2004년 5월

[4] CAN in Automation, http://www.can-cia.org

[5] PHILIPS, http://www.semiconductors.philips.com

[6] CAN in Automation, http://www.can-cia.org

[7] Karl Henrik Johansson, Martin Torngren, Lars Nielsen, "Vehicle Applications of Controller Area Network", Handbook of Networked and Embedded control Systems, Brikhauser, 2005. Invited paper

[8] Magnus Persson, Fredrik Gustafsson, "A Bluetooth-CAN Gateway", Electrical Engineering Students at Chalmers University of Technology for Sigma System AB, 2001

[9] P.Bhagwat, "Bluetooth: Technology for Short-Range Wireless Apps", IEEE INTERNET COMPUTING, May, 2001.

[10] Magnus Gustafson, Per Magnusson,

"Wireless Application Development in a Telematics Environment", Master's Degree in Computing Science Chalmers University of Technology, Sweden, 2003.

[11] D.Reilly, A. Taleb-Bendiab, "A Service-Based Architecture for In-Vehicle Telematics Systems", 22nd International Conference on Distributed computing Systems Workshops, p.741, July 2002.

[12] 김기영, 김동균, 이상정, "텔레매틱스 서비스 게이트웨이 설계 및 구현", 한국정보과학회 한국컴퓨터종합학술대회 2005, 2005년 7월

[13] 이진희, 운용익, "차세대 텔레매틱스 서비스를 위한 통합 제어 플랫폼", 한국정보처리학회 학회지, Vol.11 No.4 p.45-55, 2004년 7월

[14] 한은영, 김경호, 장정아, "텔레매틱스 단말 S/W 플랫폼 표준 아키텍쳐 설계", GIS/LBS 2004 학술대회, p. 132-138, 2004년 12월.

부 록

API 함수 리스트 및 기능

분 류	함 수 기 능
CDMA	void CDMASendMsg (struct CDMAMsg, CDMASendMsg); 원거리의 TSP(Telematics Service Provider) 및 사용자에게 메시지를 송신한다.
	void CDMAMsg CDMAReceiveMsg (void); 원거리 사용자의 제어 및 TSP로부터 메시지를 수신한다.
블루투스	void BlueSendMsg (char * SendMsg); 블루투스 통신을 이용하여 상태정보를 송신한다.
	char * BlueReceiveMsg (void); 사용자 및 핫스폿지역에서 블루투스 통신을 이용하여 제어 및 모니터링 데이터를 수신한다.
	void BlueSet (void); 블루투스 통신을 위한 초기 설정과정을 수행한다.
GPS	char * GetGps (void); GPS 수신기로부터 위치정보를 수신한다.
	void SendGps (char GpsData, struct CDMAMsg sendMsg); 원격지의 TSP에게 현재 자동차의 위치를 송신한다.
CAN	struct CANMsg GetDevice (UINT32 DeviceId); CAN통신을 이용하여 자동차 내부 장치의 상태를 모니터링 하기위해 상태정보를 얻어온다.
	struct CANMsg SetDevice (struct CANMsg SetCANMsg); CAN 통신을 이용하여 자동차 내부의 각종 장치를 제어하는데 사용한다.

분 류	함 수 기 능
CAN	char * SendStat (struct CANMsg StatCanMsg); 현재 상태정보를 송신하는데 사용한다.
	struct CANMsg GetEngin (void); 현재 엔진의 상태 정보를 CAN통신을 이용하여 얻어온다.
	void CANSet (void); CAN통신 모듈을 사용하기 위하여 통신 초기 설정을 수행한다.
메시지 파서	UINT8 * MsgParser (struct CANMsg StatCanMsg); CAN 메시지 형태의 자동차 내부 장치의 상태를 파악하기 위해 상태정보만 추출한다.
	struct CANMsg BlueToCAN (char * BlueControlMsg); 사용자로부터 수신된 제어메시지를 CAN제어 메시지 형태로 변환한다.
	char * CANToBlue (struct CANMsg statCanMsg); 자동차 장치로부터 받은 CAN메시지 형태의 정보를 사용자가 이해할 수 있는 형태인 텍스트 메시지 형태로 변환한다.
	struct CANMsg CDMAToCAN (struct CDMAMsg ReceiveMsg); CDMA 장치로부터 받은 제어 메시지를 CAN 제어메시지 형태로 변환한다.
	char * CANToCDMA (struct CANMsg StatCanMsg); CAN 메시지 형태의 상태정보를 CDMA메시지 형태로 변환한다.
기 타	char * CANIdToStr (UINT32 id); CAN으로부터 받은 장치 식별자 값을 사용자가 이해할 수 있는 형태로 변환한다.
	char * CANInfoToStr (UINT8 data); CAN으로부터 받은 상태정보를 사용자가 이해할 수 있는 형태로 변환한다.
	UINT32 StrToCANId (char * id); 사용자로부터 받은 제어메시지 중 장치명을 CAN 메시지 식별자 값으로 변환한다.
	UINT8 StrToCANInfo (char * data); 사용자로부터 받은 제어메시지 중 제어신호를 CAN 메시지 제어데이터 값으로 변환한다.
	struct CDMAMsg StrToCDMAMsg (char * receiveCDMA); 원격지의 TSP 또는 사용자에게 정보를 보내기 위해 생성한 텍스트 형태의 메시지를 CDMA 메시지 자료형으로 변환한다.

◐ 저 자 소 개 ◑

김 기 영 (Ki-Young Kim)
2003년 순천향대학교 컴퓨터공학 학과 졸업(학사)
2005년 순천향대학교 대학원 전산 학과 졸업(석사)
2005~현재 (주)센트로닉스 방송통신SOC연구소 시스템팀 연구원
관심분야 : 텔레매틱스, 홈 네트워크, 임베디드 시스템, SIP
E-mail : k71077@sch.ac.kr

김 동 균 (Dong-Kyun Kim)
1997년 금오공과대학교 공과대학 기계공학과 졸업(학사)
1996-2000년 동양그룹 (주)동양매직 가전연구소 연구원
2002년 순천향대학교 공과대학 컴퓨터공학과 졸업(학사)
2004년 순천향대학교 대학원 전산학과 졸업(석사)
2004~현재 순천향대학교 대학원 전산학과 박사과정
관심분야 : 홈 네트워크, 텔레매틱스, IP 네트워크, 임베디드 시스템
E-mail : kdk70@sch.ac.kr

이 상 정 (Sang-Jeong Lee)
1983년 한양대학교 공과대학 전자공학과 졸업(공학사)
1985년 한양대학교 대학원 전자공학과 졸업(공학석사)
1988년 한양대학교 공과대학 전자공학과 졸업(공학박사)
1988~현재 순천향대학교 정보기술공학부 교수
1999~2000년 미국 University of Minnesota 방문교수
관심분야 : 네트워크 응용, 컴퓨터 구조
E-mail : sjlee@sch.ac.kr