SDL 도구를 이용한 WTP 프로토콜의 구현 및 시험
(Implementation and Testing of the WTP Protocol using SDL Tools)

이 해 동 † 정 호 원 †† 원 유 재 ††† 임 경 식 ††††
(Haedong Lee) (Howon Jung) (Yoojae Won) (Kyungshik Lim)

요 약 본 논문에서는 SDT를 이용하여, WAP 포럼에서 제안한 WTP 프로토콜을 설계, 검증 및 구현하였다. 이를 위하여 무선 트랜잭션 프로토콜을 형식 명세 언어인 SDL로 모델링하였으며, UDP 플랫폼 상에서 동작하는 무선 트랜잭션 프로토콜 소프트웨어를 생성하기 위한 환경함수를 설계 및 구현하였다. 또한, 무선 트랜잭션 프로토콜 서비스를 사용하여 통신 응용 프로그램을 작성할 수 있도록 개발 환경을 제공하기 위해 응용 프로그래밍 인터페이스를 설계하였다. 그리고 시험 도구인 ITEX를 이용하여 구현된 프로토콜이 규격에 일치하여 동작하는지 검사하는 적합성 시험을 수행하였다. 이를 위하여 시험 언어인 TTCN으로 추상적 시험 스위트를 작성하였고 시험 언어 컴파일러를 이용하여 실행 가능한 시험 스위트를 생성하였다.

Abstract In this paper, we design, validate and implement WTP(Wireless Transaction Protocol) using SDT(SDL Design Tool). We do modeling WTP protocol by SDL(Specification and Description Language), design and implement the environment function for the interface between the SDL system and the UDP platform and design APIs(Application Programming Interface). And we do conformance testing for WTP protocol software using ITEX(Interactive TTCN Editor and eXecutor). We write ATS(Abstract Test Suite) by TTCN(Tree and Tabular Combined Notation) and make ETS(Executable Test Suite) by the TTCN compiler supplied by ITEX.

1. 서 론

월드와이드웹(World Wide Web)의 탄생으로 인터넷은 급속한 성장을 이루어 왔으며, 이와 관련된 기술의 발달로 사용자는 개인용 컴퓨터를 이용해 웹에 접근해서 자신이 원하는 정보를 얻을 수 있게 되었다. 하지만 기존의 인터넷은 고정된 유선망에 기반을 둠으로써, 공간적 제약이 따른다. 따라서, 최근 언제 어디서나 인터넷에 접속하여 다양한 정보를 이용함으로써, 기존 인터넷 환경의 공간적 제약을 극복하기 위한 무선 인터넷 서비스에 대한 연구가 활발히 진행되고 있다. 이러한 연구의 일환으로서 무선 응용 프로토콜[1]이 출현하게 되었다.

본 논문은 무선 응용 프로토콜 스택 중 트랜잭션 처리 모듈인 무선 트랜잭션 프로토콜을 구현하고 규격에 맞게 구현되었는지를 검사하는 적합성 시험에 관하여 언급한다.

전통적으로 프로토콜은 CHILL, MODULA-2, C, PASCAL, JAVA 등의 언어로 구현 및 시험되어 왔다. 그런데, 컴퓨터 통신 시스템은 두 노드 사이의 상호작용 및 정보전달을 수행하는 것으로 기능의 신뢰성을 향상시키는 것이 필요하다. 본 연구에서는 SDL(Specification and Description Language)[2,3]에 기반한 형식기법(Formal Description Technique)을 이용하여 프로토콜을 기술함으로써, 설계의 정확성을 제공하고, 자동화 도구인 SDT(SDL Design Tool)[4]를 활용하여 검증 및 구현과정을 쉽게 함을 목적으로 한다.

최근 통신 시장의 국제화로 인해, 상호 운용성이 통신 시스템의 필수적 사항으로 되어가고 있다. 적합성 시험은 상호운용성의 전제조건으로서, 프로토콜 구현물이 프

† 정 회 원 : 한국전자통신연구원 AAA정보보호연구팀 연구원
haenamu@etri.re.kr
†† 비 회 원 : 경북대학교 컴퓨터과학과
won@ccmc.knu.ac.kr
††† 종신회원 : 안철수연구소 IA 시큐리티 기술이사
yjwon@ahnlab.com
†††† 정 회 원 : 경북대학교 컴퓨터과학과 교수
kslim@knu.ac.kr
논문접수 : 2000년 11월 29일
심사완료 : 2001년 6월 4일

로토콜 규격에 일치하는지에 대해 시험함을 목적으로 한다. 특히, 적합성 시험의 방법과 체계는 ISO에 의해 ISO 9646 CTMF(Conformance Testing Methodology and Framework)[5]로 표준화되었다. 본 연구에서는 ISO 9646 part 3에서 정의한 프로토콜 시험언어인 TTCN(Tree and Tabular Combined Notation)[5]과 개발 도구인 ITEX(Interactive TTCN Editor and eXecutor)[4]를 사용하여 SDT로 구현된 무선 트랜잭션 프로토콜을 시험하였다.

본 논문의 전체적 구조는 다음과 같다. 2장에서는 무선 응용 프로토콜의 기본개념 및 무선 트랜잭션 프로토콜의 기능을 분석한다. 3장에서는 SDT를 이용한 무선 트랜잭션 프로토콜의 구현에 대하여 설명한다. 4장에서는 ITEX를 이용한 적합성 시험에 대하여 설명한다. 마지막으로 6장에서는 결론을 내리고자 한다.

2. WAP의 기본개념과 WTP의 기능 분석

2.1 WAP의 기본개념과 기술동향

본 절에서는 무선 인터넷 서비스를 지원하기 위한 기술의 표준화 동향을 살펴보고, 무선 응용 프로토콜 스택과 무선 트랜잭션 프로토콜의 구현사례를 언급한다.

현재 WAP 포럼 진영과 ME(Mobile Explorer) 진영 사이에 표준화 경쟁이 활발히 진행중이며, NTT 도코모의 i-모드는 일본내에서 상업적으로 성공을 거두고 있다. 무선 응용 프로토콜은 연결 분리 기법을 채택하여 구간마다 서로 다른 프로토콜을 사용하고 무선 환경에 최적화된 마크업 언어를 제안한다. 그림 1과 같이, 무선 응용 프로토콜은 분리된 연결을 동적으로 연동하기 위하여 무선 응용 게이트웨이(WAP gateway)를 필요로 하며, 이 게이트웨이를 통해 클라이언트는 인터넷으로 연결된다. ME는 기존 유선 인터넷 기술과의 호환성에 중점을 두며, 마이크로소프트사에 의해 주도되고 있으며,

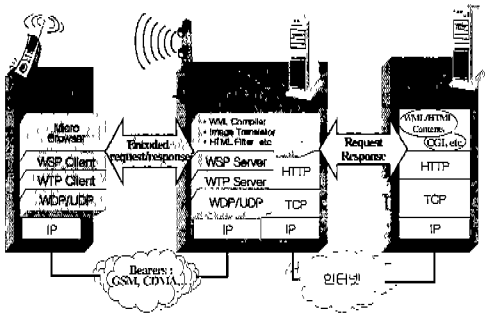

그림 1 WAP 모델

무선 응용 프로토콜의 강력한 경쟁 기술이다. i-모드는 일본내에서의 상용화란 시장의 제약으로 국제적 표준으로 채택되기에는 힘들 것이다.

무선 응용 프로토콜 스택[1]에서 각각의 계층은 상위 계층에 의해 접근될 수 있으며 다른 서비스나 응용프로그램에 의해서도 이용될 수 있다. 무선 응용 프로토콜 스택은 웹과 이동 전화 기술에 기반한 일반적인 목적의 응용 프로그램 환경을 제공하는 무선 응용 환경(WAE), 무선 세션 프로토콜(WSP), 무선 트랜잭션 프로토콜(WTP), 무선 전송 계층 보안 프로토콜(WTLS), 무선 데이타그램 프로토콜(WDP)로 구성되어 있다.

무선 트랜잭션 프로토콜 구현사례로는 WAP 포럼의 주도적 기업인 노키아, 폰닷컴, 모토로라, 에릭슨에서 배포한 실행파일 등이 있다. 그리고 캐널과 GNU 등에서 소스코드를 공개하고 있다. 대부분의 구현물들은 C, JAVA 등의 언어로 구현되었다. 본 논문에서는 형식 언어(Formal Description Language)를 사용하는 방법에 대해 제시한다.

2.2 WTP 기능 분석

본 절에서는 무선 응용 프로토콜 스택 중에서 본 연구에서 개발하는 무선 트랜잭션 프로토콜의 기능을 분석한다. 무선 트랜잭션 프로토콜은 무선 환경에 적합하도록 설계되었으며, 다음의 특징을 가진다.

첫째, 상위 계층의 요구에 따라 수행되는 트랜잭션의 종류를 세분화시켜 차별화된 서비스를 제공함으로써 무선망의 자원을 효율적으로 사용한다. 둘째, 무선 트랜잭션 프로토콜의 하위 계층인 전송계층의 비신뢰성을 해소하기 위해서, 유일한 트랜잭션 식별자와 응답, 재전송 기법 등을 통하여 프로토콜 상위 계층에 신뢰성을 제공한다. 셋째, 기존 유선망에서의 TCP 프로토콜에서와 같은 명시적인 연결 설정 및 종료를 하지 않음으로써 통신 링크 오버헤드(overhead)를 줄인다. 또한, 사용자 확인(user acknowledgement), 취소 메커니즘, 비동기 트랜잭션, 트랜잭션 식별자 검증 등의 기능을 제공한다.

2.3 계층간 통신 요소 분석

2.3.1 WTP 프로토콜의 상위 계층을 위한 서비스 프리미티브

OSI 계층 모델은 통신 시스템 구조를 위한 모델로서 한 계층의 프로토콜 개체는 상위 계층에 서비스를 제공하고 하위 계층에 서비스를 요청한다. 무선 트랜잭션 프로토콜 개체는 상위 계층에 위치한 프로토콜 사용자에게 트랜잭션 처리 서비스를 제공한다. 프로토콜 제공자와 사용자 사이의 상호작용은 서비스 엑세스 포인트(소프트웨어 포트)를 통하여, 서비스 프리미티브를 전달하

여 이루어진다.

무선 트랜잭션 프로토콜 규격서에서는 TR-Invoke, TR-Result, TR-Abort로 구성된 세 가지 서비스 프리미티브를 정의하고 있다. TR_Invoke는 프로토콜 사용자가 새로운 트랜잭션을 시작하기 위해서 사용되며. TR_Result는 프로토콜 사용자가 이미 시작된 트랜잭션 결과를 원격 호스트의 프로토콜 동등 개체에 송신하기 위해서 사용된다. 마지막으로 TR_Abort는 프로토콜 사용자가 수행중인 트랜잭션을 취소시키기 위해서 사용된다. 그리고 각각의 서비스 프리미티브는 요구(request), 통지(indication), 응답(response), 확인(confirm)의 세부화된 종류를 가진다.

2.3.2 WTP 프로토콜 데이타 유닛의 구조 및 인코딩

프로토콜 동등 개체들은 프로토콜 데이타 유닛을 교환함으로써 통신한다. WTP 프로토콜 데이타 유닛의 실제 전송은 IP망에서 UDP의 서비스를, non-IP망에서 WDP의 서비스를 이용하도록 설계되었다. WTP의 프로토콜 데이타 유닛은 표 1와 같이 정의되어 있다.

표 1 WTP PDU Types

PDU type	PDU code
for concatenation	0x00
Invoke	0x01
Result	0x02
Ack	0x03
Abort	0x04
Segmented Invoke	0x05
Segmented Result	0x06
Negative Ack	0x07

Invoke PDU는 하나의 트랜잭션을 시작할 때 사용된다. Result PDU는 Invoke PDU에 대한 결과로서 돌려받는 메시지이다. Ack PDU는 Invoke 또는 Result PDU의 수신에 대한 응답 메시지이다. Abort PDU는 트랜잭션의 강제적 취소를 위해서 사용된다. 그리고 분할 및 재결합(Segmentation And Re-assembly) 기능을 지원하기 위해서 Segmented Invoke PDU, Segmented Result PDU 및 Negative Acknowledgement PDU를 정의하고 있다.

프로토콜 데이타 유닛 코드가 0x00일 경우, 여러 개의 프로토콜 데이타 유닛이 하나의 서비스 데이타 유닛에 포함되어 있음을 의미한다. 서비스 데이타 유닛으로부터 각각의 프로토콜 데이타 유닛을 추출하는 디코딩을 거친 후, 각각의 프로토콜 데이타 유닛은 프로토콜 논리 기능을 수행하는 해당 모듈에 전달되어 처리된다.

3. SDT를 이용한 WTP 프로토콜 구현

3.1 형식 기법의 소개

본 절에서는 SDT를 WTP 프로토콜 개발에 적용한 연구내용을 설명하기 전에, SDT 및 SDL에 대한 간략한 설명과 국내의 형식 기법과 관련된 연구사례를 언급한다.

SDT는 Telelogic사에서 제공하는 개발 도구로서, SDL 편집, 구문적/의미적 오류의 검출, C 소스 코드의 생성, 다양한 시뮬레이션 방법들을 제공한다. SDL은 통신 시스템을 기술하기 위한 언어로서 ITU-T에서 개발되었고, 구조적 개념, 설계의 최적화, 구현의 유연성, 및 설계자 상호간의 이해 증진을 위한 기법을 제공함으로써 프로토콜의 동작을 기능적으로 명확하게 표현할 수 있다. 또한, SDL은 정형적 표현 기법으로 인하여 설계 단계에서의 시스템 분석 방법을 제공하고 있다.

국내의 SDL 관련 연구사례로는 SDT를 이용한 WLL 프로토콜 검증에 대한 연구가 있었다[6]. 적합성 시험사례로는 ATM망에 연결되는 단말기에 대해 적합성 시험 방법과 결과 분석 방법에 대한 연구가 있었다[7]. 차세대 지능망의 INAP 프로토콜 적합성 시험 계열 생성을 위해 SDL을 사용하여 프로토콜을 명세하고 이것으로부터 시험 계열을 생성하기 위한 최적화 기술에 의한 방법을 제안되었다[8]. 이 기종 간의 LAN을 이용하여 구현된 적합성 시험 시스템을 이용하여 CCITT No. 7 TCAP 프로토콜에 대한 적합성 시험이 수행되었다[9].

3.2 전체 구현 모듈 개요

먼저 SDT를 이용하여 WTP 프로토콜을 구현함에 있어서, 전체 구현된 모듈에 대한 소개와 설명을 간략히 제시하고 다음절부터 각각의 모듈을 구현하는데 필요한 설계 및 방법을 설명한다. 그리고 구현시 세부사항은 부록에 첨부하였다.

그림 2는 SDT를 이용하여 시스템을 구현할 때 소프트웨어의 구조 및 각 모듈의 역할을 설명하는 그림이다. SDL 시스템은 상태와 상태천이로 설계된 추상적 프로토콜 명세를 SDT의 코드 생성기에 의해 C 소스 코드로 변환된 모듈이며 운영체제, 파일 시스템, 하드웨어, 네트워크 플랫폼 등의 물리적 환경에 대한 어떠한 정보도 가지고 있지 않다.

SDT로 구현되는 통신 시스템에서는 실제 메시지의 수신 및 송신 그리고 인코딩/디코딩을 지원하는 환경함

수(environment function)[3,4]가 필요하다. 즉 SDL 시스템과 물리적 환경사이의 인터페이스의 역할을 수행하는 환경함수를 구현하고 이 환경함수 모듈과 SDL 시스템을 함께 컴파일 및 링크시킴으로써, 완전한 실행코드를 생성하였다.

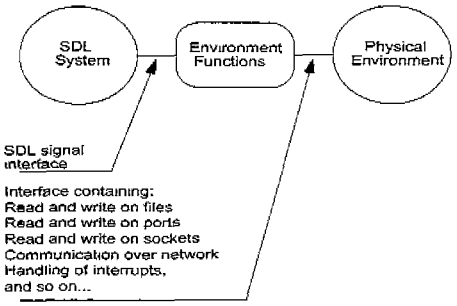

그림 2 SDT로 구현된 소프트웨어 구조[4]

한편, 통신 응용 프로그래머에게 편리한 개발 환경을 제공하기 위해서 응용 프로그래밍 인터페이스(Application Programming Interface)를 설계하였다.

3.3 SDL 시스템 설계

본 절에서는 무선 트랜잭션 프로토콜의 설계에 필요한 데이타 타입, 메시지 전달을 위한 시그널, 시그널 라우트, 채널 등의 정의를 언급하고, 프로토콜 개체를 형성하는 모듈들에 대해서 설명한다. 무선 트랜잭션 프로토콜을 위한 SDL 명세는 시스템, 블록 및 프로세스 명세에 의해 계층적으로 표현된다.

3.3.1 시스템 명세

WTP 프로토콜 위한 시스템 명세는 그림 3와 같이, WTP 프로토콜 개체의 동작이 설계되는 하나의 블록 명세와 시스템 외부와의 인터페이스인 상하위 계층과의 인터페이스를 의미하는 TRsapS 채널과 UsapS 채널을 가지게끔 설계되었다. 부록의 표 1에서 이러한 채널상에서 전달되는 시그널 및 시그널의 묶음인 시그널 리스트의 정의를 첨부한다.

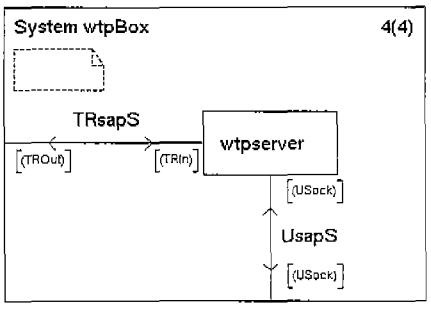

그림 3 WTP 서버의 시스템 명세

3.3.2 블록명세

WTP 프로토콜 블록 명세는 무선 트랜잭션 프로토콜의 논리기능을 수행하는 요구자 프로세스(Initiator) 및 응답자 프로세스(Responder)로 구성된 트랜잭션 수행 프로세스와 클라이언트 관리 프로세스(Client_Manager) 및 트랜잭션 관리 프로세스(TR_Manger)인 멀티플랙싱 프로세스로 구성되어 있으며, 이들 프로세스간의 상호작용 및 협조를 통해서 프로토콜 동작이 이루어진다.

트랜잭션 수행 프로세스는 트랜잭션의 수만큼 동적으로 생성된다. 여러 동등 개체 및 프로토콜 사용자에 의한 시그널은 각각의 해당 트랜잭션 수행 프로세스에 전달되어야 한다. 이러한 멀티플랙싱을 위해서 클라이언트 관리 프로세스와 트랜잭션 관리 프로세스를 추가적으로 설계하였다. 블록 명세 내에서 SDL 프로세스의 구조는 시그널 전달 흐름 및 프로세스의 역할에 따라 계층적 특징을 가진다. 즉, 상하위 계층으로부터 수신된 이벤트는 먼저 클라이언트 관리 프로세스를 거쳐 해당 트랜잭션 관리 프로세스에 전달되고, 다시 해당 트랜잭션 수행 프로세스에 배분되면, 이 응답자/요구자 프로세스가 프로토콜 논리기능을 수행한다. 그림 4에서 사용되는 시그널 및 시그널 라우트는 부록의 표 1, 표 2, 표 3에 싣는다.

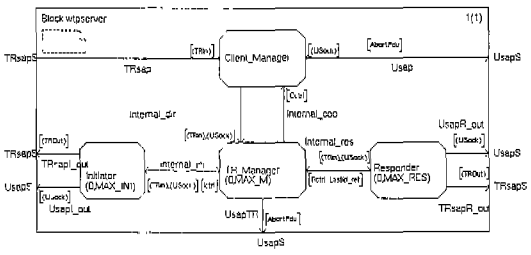

그림 4 서버측 블록 명세

3.3.3 프로세스 명세

가. 클라이언트/트랜잭션 관리 프로세스

(1) 상·하위 계층으로부터 발생한 시그널의 처리

클라이언트 관리 프로세스는 시스템이 초기화될 때 정적으로 생성되어 시스템이 종료할 때까지 존재하며, 각각의 원격호스트에 있는 클라이언트에 대해 별도의 트랜잭션 관리 프로세스를 동적으로 생성하여 개별 클라이언트와 연관시키는 역할을 수행한다. 그리고, 생성된 트랜잭션 관리 프로세스는 하나의 클라이언트에 속

한 각각의 트랜잭션들에 대해 요구자 혹은 응답자 프로세스를 동적으로 생성하여 개별 트랜잭션과 연관시키는 역할을 수행한다.

상·하위 계층으로부터 발생한 시그널이 해당 요구자 혹은 응답자 프로세스에 전달되어 무선 트랜잭션 프로토콜의 논리기능을 수행하기 위해서, 클라이언트 관리 프로세스와 트랜잭션 관리 프로세스는 시그널의 적절한 멀티플렉싱을 수행한다. 이를 위해서 상위 프로세스들은, 하위 프로세스가 동적으로 생성될 때, 트랜잭션 관련정보와 생성된 프로세스의 정보를 포함하는 매핑 테이블을 관리하여 멀티플렉싱에 이용한다. 좀더 자세히 설명하면, 클라이언트 관리 프로세스는 클라이언트의 주소/포트 혹은 핸들(handle) 범위에 기초하고, 트랜잭션 관리 프로세스는 트랜잭션 식별자(transaction identifier) 혹은 핸들 값에 기초하여 하위 프로세스 식별자(process identifier)를 검색할 수 있다. 부록의 그림 1에서 클라이언트 관리 프로세스가 유지하는 매핑 테이블 정의를 첨부하였다.

(2) 프로세스의 종료를 위한 내부시그널의 처리

트랜잭션 관리 프로세스와 요구자 및 응답자 프로세스는 수행되는 트랜잭션의 유무에 따라 동적으로 생성 및 종료된다. 이때 매핑테이블에 트랜잭션 및 프로세스의 정보가 저장되어 있다. 프로세스가 종료하면, 매핑테이블의 수정이 필요하게 되었다. 종료를 알리는 내부시그널을 도입함으로써 이러한 문제를 해결하였다. 즉, 하위 프로세스는 트랜잭션 수행이 정상 혹은 비정상 종료될 때, 상위 프로세스에게 종료 시그널을 전달하고 자신을 소멸시킨다. 상위 프로세스는 종료 시그널에 의해 소멸된 프로세스에 대한 엔트리를 매핑 테이블에서 제거하고 수정한다. 이를 통해서 소프트웨어 자원인 매핑테이블의 재사용과 프로세스 관리를 통한 다중 트랜잭션 및 비동기 트랜잭션의 특징을 제공하게 한다.

나. 요구자/응답자 프로세스

요구자 및 응답자 프로세스는 차별화된 트랜잭션 서비스, 재전송 메커니즘, 사용자 확인, 트랜잭션 취소, 트랜잭션 식별자 검증, 각종 부적절한 필드 값에 의한 에러의 처리 등 무선 트랜잭션 프로토콜의 논리 기능을 실제 수행하는 모듈이다. 그림 5는 프로세스 명세의 한 예로서, 트랜잭션 식별자 검증을 위한 응답자 프로세스의 다이어그램을 보여준다.

실제 메시지의 송수신 및 인코딩/디코딩을 담당하는 환경함수는 수신한 메시지를 시그널로 변환시켜 SDL 시스템으로 보낸다. 그러나 환경함수에서 수신된 메시지의 타입을 추출할 수 없다면 명시적 시그널 포맷을 선택하여 시그널화 할 수 없었다. 에러 PDU를 위한 에러 시그널을 새로이 정의하여 이러한 문제를 해결하였다. 그림 6에서 보는 바와 같이, 에러 시그널은 <주소, 포트> 및 트랜잭션 식별자를 인자로 가지게 하였다. 그 이유는 클라이언트 관리 프로세스는 <주소, 포트>로 해당 트랜잭션 관리 프로세스를 찾을 수 있고, 트랜잭션 관리 프로세스는 트랜잭션 식별자로 프로토콜 논리 기능을

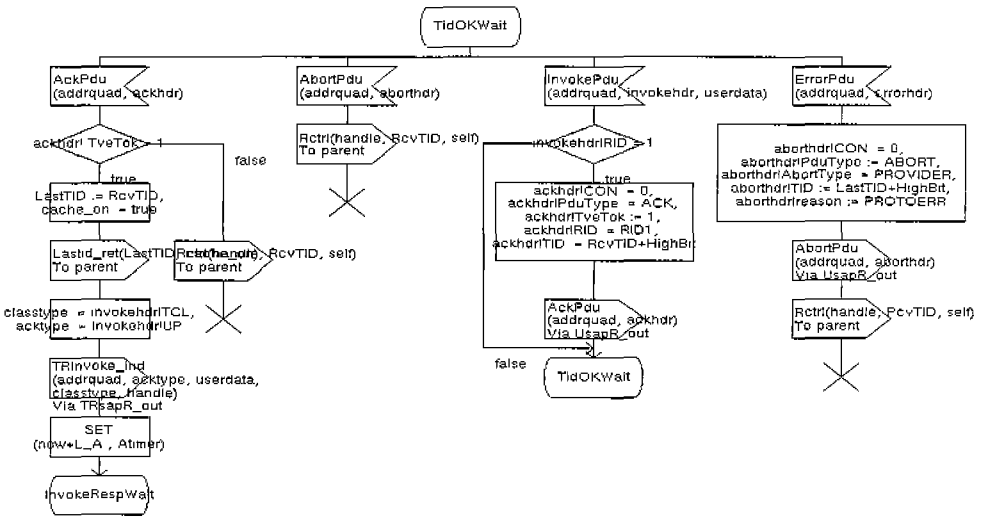

그림 5 TidOKWait 상태에서의 상태천이

수행하는 해당 요구자 혹은 응답자 프로세스를 찾을 수 있기 때문이다.

```
Newtype ErrorHDR
        TID TID_TYPE;
EndNewtype;
Signal ErrorPdu(AddrQuadruplet, ErrorHDR);
```

그림 6 ErrorHDR 구조체 및 ErrorPdu 시그널

또한, 환경함수에서 검사하지 않은 헤더의 다른 필드 즉, 메시지 타입 외의 필드에 대한 검사는 트랜잭션 관리 프로세스에 의해 수행하게 하였다. 유효하지 않은 필드가 존재하면, ErrorPdu 시그널로 변환하여 해당 트랜잭션 수행 프로세스에 전달한다.

3.4 WTP 프로토콜 검증

WTP 규격서를 분석하여 SDL로 WTP 프로토콜 시스템을 설계한 후 SDL 명세 문서를 검증도구인 SDT의 Analyzer로 구문적 의미적인 오류를 검사함으로써

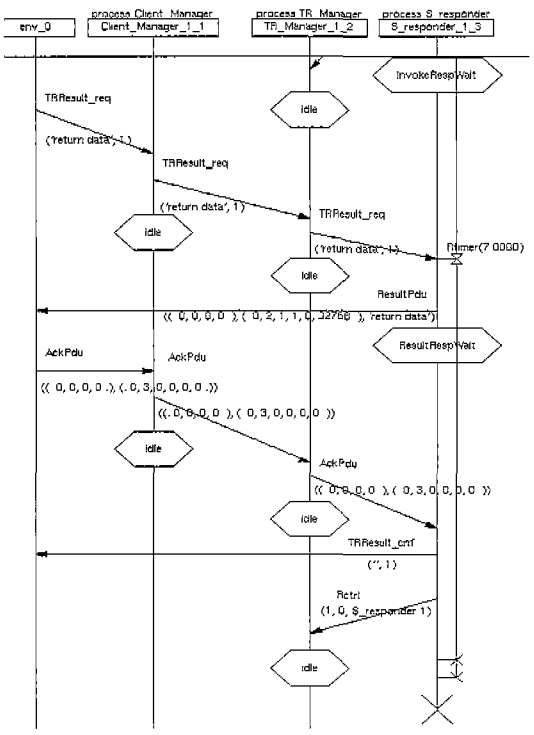

그림 7 MSC - 클래스 2 트랜잭션 시그널링 동작

SDL로 명세된 WTP 프로토콜의 시스템, 블록, 프로세스에 대한 정확성을 분석한다[4]. 시그널이 시스템 다이어그램과 블록 다이어그램에 정의되어 있지 않거나, 정의된 것과 다르게 입출력이 뒤바뀐 경우, 잘못된 채널 및 라우터의 지정과 다중 수신 등의 오류를 찾아낸다.

구문적 및 의미적 오류 검사를 한 다음, 시뮬레이션을 수행하고 MSC(Message Sequence Chart)를 생성하여 전송되는 파라미터와 이벤트 발생에 따라 천이하는 상태가 정확한지 검증을 수행한다. 그림 7은 프로토콜 검증 과정에서 산출된 MSC로서, WTP 프로토콜이 응답자로 동작할 때 클래스 2 트랜잭션이 수행되는 과정의 일부를 보여준다.

3.5 환경함수 구현 및 실행코드 생성

3.5.1 환경함수의 역할 및 구조

구현된 환경함수는 xInitEnv, xCloseEnv, xInEnv, xOutEnv 함수로 구성되어 있다. 본 논문에서는 서버측 환경함수에 초점을 맞추어 설명한다. 표 7은 무선 트랜잭션 프로토콜 소프트웨어를 구현하는데 필요한 각각의 환경함수 모듈에 대해 설명한다.

- xInitEnv : 시스템이 처음 시작할 때 대표소켓(9201포트)을 만든다.

- xInEnv : 프로토콜 동등 개체로부터 메시지를 혹은 WTP 사용자로부터 서비스 프리미티브 이벤트를 수신하면, xGetSignal 함수 호출을 통하여 해당타입의 빈 시그널 데이타 영역을 만든다. 그리고 타입에 의거해서 버퍼의 내용을 디코딩하여 시그널 데이타 영역에 복사를 한다. 마지막으로 SDL_ Output 함수 호출을 통하여 SDL 시스템에 전달한다. xInEnv 함수의 구조는 부록의 그림 2와 같다.

- xOutEnv : SDL 시스템에서 전달된 시그널을 타입에 따라 실제 메시지로 인코딩하여 출력 버퍼에 복사하고 소켓에 대한 쓰기 함수 호출을 통하여 실제 물리적 환경인 네트워크나 WTP 사용자 프로세스에게 출력 버퍼의 내용을 전송한다. 마지막으로 xReleaseSignal 함수 호출을 통하여 xOutEnv 함수의 인자로 넘겨받은 시그널 데이타 영역을 해제한다. xOutEnv 함수의 구조는 부록의 그림 3과 같다.

- xCloseEnv : 소켓에 대한 닫기 함수로 대표소켓을 해제한다.

3.5.2 메시지 포맷

통신 응용 프로그램과 무선 트랜잭션 프로토콜 소프트웨어는 IPC를 통하여 서비스 프리미티브 이벤트를 교환한다. 그러므로 환경함수에서는 응용 프로그램과 TCP 연결 설정을 제공하고, TCP 채널을 통해 교환되는 메

시지 포맷에 따라 이벤트를 인코딩/디코딩한 후, 송수신을 수행한다.

```
struct msgHeadType {
        int type;
        int acktype;
        size_t data_size;
        size_t exit_size;
        int classtype;
        int abortcode;
        int handle;
        int dis_flag;
};
```

그림 8 IPC를 위한 메시지 포맷

서비스 프리미티브의 타입과 그 인자에 기초하여 메시지의 포맷을 정의하였다. 메시지의 형태가 단순한 경우에 설계 및 구현이 용이하다. 이를 위해서 메시지의 포맷을 하나로 고정하였다. 그림 8은 메시지의 구조를 보여준다. 사용되는 구조체의 필드는 프리미티브 종류에 따라 달라진다. 예를 들어, TRResult_req 프리미티브에서는 type, data_size, handle 필드만 사용된다.

3.5.3 응용 프로그래밍 인터페이스 함수 설계

본 연구에서 설계 및 제안한 무선 트랜잭션 프로토콜 시스템은 운영체제상에서 독립된 프로세스로 동작하며, IPC를 통해 통신 응용 프로그램과 메시지를 상호 교환한다. 한편, 통신 응용 프로그래머에게 편리한 개발 환경을 제공하기 위해서 IPC 세부사항을 숨길 필요가 있다[10,11]. 이를 위하여 IPC 관련 사항을 포함하는 응용 프로그래밍 인터페이스를 설계하였다. 응용 프로그래밍 인터페이스는 응용 프로그램 모듈과 프로토콜 모듈사이에 있는 가상적 라이브러리 함수이며, 부록의 표 4와 같다. 그림 9는 응용 프로그램 개발자가 클래스 2로 트랜잭션 서비스를 요청할 때 응용 프로그래밍 인터페이스의 사용을 보여준다.

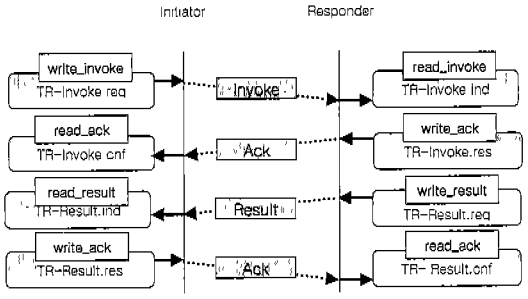

그림 9 클래스 2 트랜잭션

4. ITEX를 이용한 WTP 프로토콜 적합성 시험

4.1 적합성 시험 환경

본 절에서는 무선 트랜잭션 프로토콜의 시험 환경에 대해서 언급한다.

무선 트랜잭션 프로토콜에 대한 적합성 시험 환경은 시험대상시스템(System Under Test)과 하위시험기(Lower Tester)가 UDP 전송계층에 의해 연결된 형태이고 시험대상프로토콜 (IUT : Implementation Under Test)은 시험대상시스템에 있는 무선 트랜잭션 프로토콜이다. 적합성 시험에 응용되는 시험 기법으로 시험에 필요한 접근방식 및 IUT와의 협조관계에 따라 지역(local), 분배(distributed), 조정(coordinated), 원격(remote), ferry 방법 등이 있다[12].

그림 10과 같이, 본 연구에서는 무선 트랜잭션 프로토콜의 상위 서비스 경계면에 PCO(Point of Control and Observation)를 갖지 않는 원격적 방법을 사용하였다. 일반적으로 원격적 시험방법에서는 상위시험기(Upper Tester)를 갖지 않는다. 본 시험에서는 응용 프로그램이 상위 시험기의 역할을 수행한다. 하위시험기는 무선 트랜잭션 프로토콜 떨어진 시스템에서 UDP로 전달되는 메시지를 관찰하여 무선 트랜잭션 프로토콜 구현물의 동작에 대한 적합 판정을 내린다.

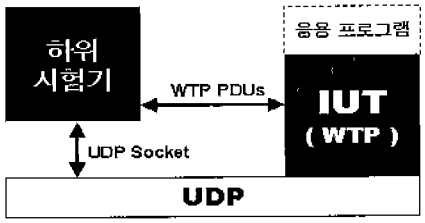

그림 10 적합성 시험 환경

4.2 추상적 시험 스위트(Abstract Test Suite) 생성

모든 무선 응용 프로토콜 규격서에는 여러 구현물 간의 상호 운용성을 보장하기 위해서 프로토콜 기능의 집합을 정의한 정적 적합성 요구사항을 제시하고 있으며, 본 연구에서는 상호 운용성을 위한 필수적 기능 및 일부 구현선택사항을 간추려 부록의 표 5와 같이 PICS (Protocol Implementation Conformance Statement) [12]를 제시하고 구현하였으며, 이러한 PICS를 기초로

추상적 시험 스위트(ATS)를 작성하였다. 추상적 시험 스위트는 실행 가능한 시험기 소프트웨어는 아니며, 어떤 기능을 시험하기 위해서 송수신되는 메시지의 순서를 추상적으로 정의한 것이다.

그림 11은 무선 트랜잭션 프로토콜에 대한 시험 스위트의 구조를 나타내고 있다. 요구자와 응답자 각각에 대해 시험 그룹으로 분리하고, 다시 각각의 시험 그룹에 대해 트랜잭션 클래스별로 시험 그룹으로 나누었다. 각각의 트랜잭션 클래스, 사용자 확인, 재전송 메커니즘, 트랜잭션 취소, 무효한 필드를 포함하는 패킷의 에러 처리, TID 검증, 중복 패킷 처리에 대해, 유효동작시험(valid behaviour)[5], 무효동작시험(invalid behaviour)[5], 동등 개체사이에 발생할 수 없는 프로토콜 데이타 유닛의 교환에 대한 시험(inopportune behaviour)[5]으로 구성하여 시험 항목을 정의하였다. 그림 11의 괄호 안의 숫자는 시험 항목의 개수로 응답자 및 요구자 모듈을 시험하기 위해 각각 61개와 32개의 시험 항목을 정의하였다.

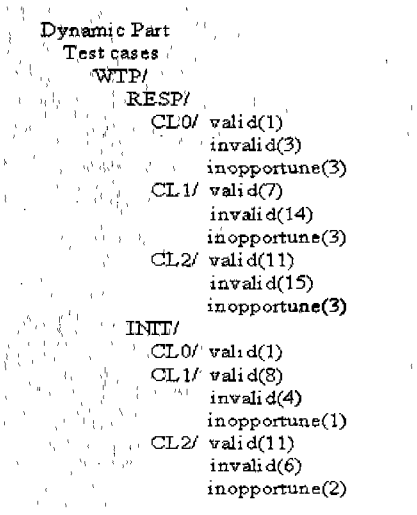

그림 11 시험 스위트 구조

적합성 시험 스위트는 여러 개의 시험 항목으로 구성되어 있는데, 이를 효율적으로 수행하기 위해서는 시험 항목을 모듈화할 필요가 있다. 일반적으로 시험 항목은 프리엠블(preamble), 시험 본체(test body), 포스트엠블(postamble)로 구성된다. 그림 12의 시험 항목은 그림 9의 메시지 흐름과 같은, 클래스 2 Hold-on Acknowledgement 트랜잭션의 동작을 시험하기 위한 것이다. 그림 13의 프리엠블은 그림 12의 시험 항목에서 시험 본체의 수행 전에 필요한 트랜잭션 식별자 검증 과정을 하나의 시험 단계로 묶은 것이다. 마지막으로, 그림 14의 포스트엠블은 시험 본체가 끝나고 일정시간을 기다린 후, IUT로부터의 무반응을 검사함으로써 시험 본체의 판정을 확실히 보장하기 위한 시험 단계로서 작성되었다.

Test Case Dynamic Behaviour

Test Case Name BV_00
Group WTP/RESP/CL2/MAND/
Purpose Hold on Acknowledgement (Class 2)
Configuration
Default
Comments
Selection Ref
Description .

Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		+PREAMBLE_CL1			
2		L1 ! INVOKE	invoke1_class2		
3		L1 ? ACK	ack1		
4		L1 ? RESULT	result1		
5		L1 ! ACK(ACK TID = '0001'O)	ack1		
6		+TS_TIMEOUT			
7		L1 ? OTHERWISE		FAIL	
8		L1 ? OTHERWISE		FAIL	

Detailed Comments

그림 12 Hold-on Acknowledgement 시험 항목

Test Step Dynamic Behaviour

Test Step Name PREAMBLE_CL1
Group
Objective
Default .
Comments .
Description

Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		L1! INVOKE	invoke_class1		invalid TID
2		L1? ACK	ack_TveTok		Tve
3		L1! ACK(ACK TID = '0000'O, ACK TveTok ='1'B)	ack		Tok
4		L1? ACK	ack		

Detailed Comments

그림 13 TID 검증을 유발하는 시험 단계 - preample

Test Step Dynamic Behaviour

Test Step Name TS_TIMEOUT
Group
Objective TIMOUT for any PDU received
Default .
Comments
Description

Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		START T_Timer			
2		?TIMEOUT T_Timer		PASS	
3		L1? OTHERWISE		FAIL	

Detailed Comments T_Timer is not WTP timer just for test

그림 14 Timeout을 처리하는 시험 단계 - postample

4.3 실행 가능한 시험 스위트(Executable Test Suite) 생성

추상적 시험 스위트를 적합성 시험기에서 수행시키기 위하여, TTCN 컴파일러를 사용해서 실행 가능한 시험

스위트(ETS)로 변환시켜야 한다[4]. ITEX의 C 소스 코드 생성기는 추상적 시험 스위트를 ANSI-C 코드로 변환하며, 변환된 모듈을 TTCN RTB(RunTime Behavior)라고 한다. RTB는 시험하려는 시스템에 대한 물리적 정보를 포함하지 않기 때문에 각종 라이브러리 파일(static help functions), 시험 지원 모듈(support function), 시험 응용 모듈(adaptor) 등과 함께 컴파일되어 시험을 수행하기에 적당한 형태로 변환된다. 시험 응용 모듈의 함수는 사용되는 프로토콜, 호스트 머신, 시험 장비에 의존적인 부분을 담당하며, RTB를 시험하려는 물리적 시스템에 적용하기 위해서 시험 수행자가 직접 구현해야 할 모듈이다. 즉, 시험 응용 모듈은 RTB와 IUT사이의 실제 전송 서비스를 제공하기 위해서, 추상적 PCO와 타이머의 구현 및 인코딩/디코딩과 메시지 송수신을 담당한다.

시험 지원 모듈 및 시험 응용 모듈을 작성하기 위해서, ITEX에서는 그림 15와 같은 GCI(Generic Compiler Interpreter) 인터페이스 모델을 사용한다. GCI 인터페이스는 표준화된 함수의 집합으로서, 이러한 함수의 호출과 함수인자를 통한 정보전달, 함수의 반환값 등으로 RTB 모듈과 시험 응용 모듈간의 통신이 이루어진다[4]. GCI 인터페이스에는 관리(management) 인터페이스, 동작(behavior) 인터페이스, 작동(operational) 인터페이스로 구성된다. 본 연구에서는 관리 인터페이스를 사용하여 시험 실행을 초기화하고 관리하는 시험 지원 모듈을 구현하였다. 작동 인터페이스는 시험자가 직접 구현해야 할 시험 응용 파일의 일부분으로 메시지 송수신 및 타이머의 동작 등을 수행한다. 또한, 인코딩/디코딩 함수를 구현하기 위해서, value 인터페이스를 사용하였다.

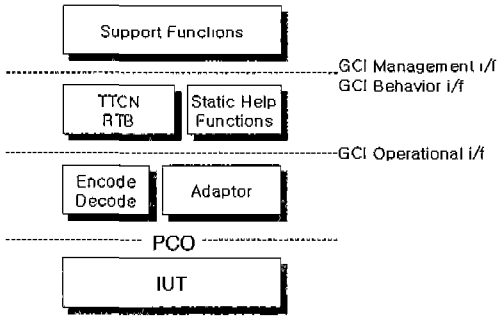

그림 15 ETS의 구조 및 GCI 인터페이스 모델[4]

4.4 시험 결과 분석

본 연구의 시험은 무선 트랜잭션 프로토콜의 필수적 기능과 구현 선택사항인 비동기적 트랜잭션에 대한 적합성 시험을 목적으로 하였다. 그림 16은 응답자의 클래스 2 트랜잭션 시험 그룹 중에서, Hold-on Acknowledgement의 기능을 시험하는 유효동작시험의 수행 결과를 보여준다. 먼저, 시험기 소프트웨어는 무선 트랜잭션 프로토콜 구현물에 Invoke 메시지를 보낸 후, 구현물로부터 Ack 메시지와 Result 메시지를 순서대로 전달받기를 기다린다. 시험기 소프트웨어는 수신된 메시지의 헤더 필드들에 대해 유효성을 검사하며, 구현물로부터 순서대로 유효한 메시지를 받으면 Ack 메시지를 구현물에 보낸다. 마지막으로, 시험기 소프트웨어는 구현물이 시험 항목에 대해 제대로 동작하였으므로 PASS 판정을 내렸다.

모든 정의된 시험 항목에 대해, 시험 목적과 일치할 때까지 시험 수행, SDL 문서 첨가 및 수정, 재컴파일, 시험 실시의 반복적 과정을 수행하였다. 이로써, 본 WTP 프로토콜 구현물의 적합성 및 타당성이 제시되었다.

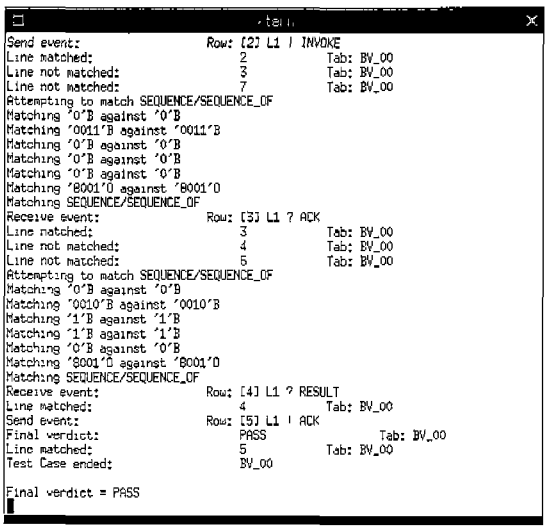

그림 16 Hold-on Acknowledgement를 위한 시험 항목 실행

한편으로, 하나의 시험기를 가지는 표준 TTCN를 사용함으로써, 병렬성을 가지는 비동기적 다중 트랜잭션을 위한 시험 항목의 작성이 불가능하였다. 따라서, 다중 트랜잭션은 TTCN으로 시험 항목을 작성하지 않고 하나의 서버와 여러 개의 클라이언트를 동시에 수행시킴으로써 서버가 병렬적으로 여러 트랜잭션을 처리하는지 검사하였다. 향후, 다중 트랜잭션에 대한 적합성 시험을 보완

하기 위해서, 멀티 파티(multi-party) 시험에 적합한 병렬 TTCN을 이용한 시험 항목의 작성이 필요하다.

5. 결 론

현재, 무선 인터넷 서비스에 대한 요구가 급증하고 있다. 이를 수용하기 위해서 무선환경에 적합한 경량의 통신 소프트웨어의 개발이 필요하다. 일반적인 컴퓨터 통신 소프트웨어는 두 노드 사이의 상호작용 및 정보전달을 수행하는 것으로 기능의 신뢰성 및 정확성을 향상시키는 것이 중요하다. 이와 관련하여 통신 프로토콜의 신뢰성, 효율성 그리고 유지보수의 용이성을 획득하기 위한 프로토콜 공학적 접근방식이 대두되었다.

따라서 본 연구는 형식 기법에 기반한 SDT와 ITEX를 이용하여, 무선 응용 프로토콜 스택 중에서 무선 트랜잭션 프로토콜을 설계 및 구현하였으며, 상호운용성의 전제 조건인 적합성 시험을 수행하였다. 즉, 프로토콜의 설계단계에서부터 검증단계, 구현단계 그리고 적합성 시험단계까지 형식 기법을 적용하여, 무선 트랜잭션 프로토콜 소프트웨어의 신뢰성 및 정확성을 높였다.

향후 과제로서 첫째, 현재의 무선 트랜잭션 프로토콜 소프트웨어는 모든 필수 기능을 포함한다. 앞으로, 필수 기능뿐만 아니라 구현선택사항에 대해서도 추가 설계 및 구현되어야 하겠다. 둘째, 적합성 시험을 위해서 표준 TTCN을 사용함으로써, 다중 트랜잭션에 대한 적합성 시험이 불충분하였다. 멀티 파티 프로토콜을 시험하는데 적합한 병렬(concurrent) TTCN[13]를 사용하여 다중 트랜잭션에 대한 적합성 시험이 보충되어야 하겠다. 셋째, 본 논문에서 구현된 프로토콜의 우수성을 위한 더욱 객관적인 근거를 제공하기 위해서, 본 시험 스위트를 다른 제 3의 구현물에 적용하여 그 결과를 비교 분석할 필요가 있다.

참 고 문 헌

[1] WAP Forum, "WAP Version 1.2 Specification, "URL: http://www.wapforum.org/

[2] Jan Ellsberger, Dieter Hogrefe, Amardeo Sarma, "SDL Formal Object-oriented Language for Communicating Systems," Prentice Hall, 1997.

[3] 이해동, 최윤석, 임경식, "SDL을 이용한 무선 트랜잭션 프로토콜의 설계 및 구현", 통신정보합동학술대회(JCCI2000)발표논문집, pp. 867-870, 2000.

[4] Telelogic AB, "Telelogic Tau 3.5 Manual," 1999.

[5] Berd Baumgarten, Alfred Giessler, "OSI Conformance Testing Methodology And TTCN," Elsevier Science, 1994.

[6] 김경민, 남영호, 박재홍, 노철우 이병길, 한운영, "SDT를 이용한 WLL 프로토콜 검증", 한국정보과학회춘계학술발표논문집, 제26권, 제1호, pp. 464-466, 1999.

[7] 장동원, "ATM망에서 프로토콜(SSCOP)적합성 시험 방법 및 결과 분석 방법", 한국통신학회논문지, 제22권, 제10호, pp. 2348-2356, 1997.

[8] 김상기, 김성운, 정재윤, "형식기술기법에 의한 AIN 프로토콜 적합성 시험 계열 생성", 한국정보처리학회논문지, 제4권, 제2호, pp. 552-562, 1997.

[9] 윤재일, 노승환, 조현윤, 김덕진, "LAN을 이용한 적합성 시험 시스템의 구현과 No.7 TCAP에의 적용에 관한 연구", 전자공학회논문지, 제29권, 제8호, pp. 629-636, 1992.

[10] 정호원, 오연주, 최윤석, 임경식, "SDT를 이용한 무선 트랜잭션 프로토콜의 프로토타입 구현", 한국정보과학회추계학술발표논문집, pp.349-351, 2000.

[11] 최윤석, 김기조, 임경식, "무선 응용 프로토콜 구현 기술", 한국정보과학회정보통신연구회정보통신기술지, 제14권, 제1호, pp. 16-31, 2000.

[12] 한국전자통신연구원(ETRI), "정보통신 프로토콜 공학", 1997.

[13] ITU, "ITU-T Recommendation X.292:OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications -The Tree and Tabular Combined Notation (TTCN)," 1998.

부 록

표 1 시그널 및 시그널 리스트

시그널리스트	설 명	
USock	데이타 전송 계층과의 상호작용을 표현하는 시그널의 모음	
	시그널	InvokePDU, ResultPDU, AckPDU, AbortPDU, ErrorPDU
TRout	WTP 사용자에게 프로토콜 이벤트를 전달하기 위한 시그널의 모음	
	시그널	TRInvoke_cnf, TRInvoke_ind, TRResult_cnf, TRInvoke_ind, TRAbort_ind
TRin	WTP 계층에게 서비스를 요청하기 위한 시그널의 모음	
	시그널	TRInvoke_req, TRInvoke_res, TRResult_req, TRInvoke_res, TRAbort_req

표 2 채널과 연결되는 시그널 라우트

시그널 라우트	설 명
Usap	시스템 명세의 UsapS 채널과 연결
	Client_Manager 프로세스가 USock 시그널리스트에 명시된 시그널을 하위 계층으로부터 전달받는 경로
TRsap	시스템 명세의 TRsapS 채널과 연결
	Client_Manager 프로세스가 TRin 시그널리스트에 명시된 서비스 프리미티브 시그널을 상위 계층(WTP 사용자)으로부터 전달받는 경로
UsapI_out	시스템 명세의 UsapS 채널과 연결
	Initiator 프로세스가 USock 시그널리스트에 명시된 시그널을 하위 계층에게 전달하는 경로
TRsapI_out	시스템 명세의 TRsapS 채널과 연결
	Initiator 프로세스가 TRout 시그널리스트에 명시된 서비스 프리미티브 시그널을 상위 계층(WTP 사용자)에게 전달하는 경로
UsapR_out	시스템 명세의 UsapS 채널과 연결
	Responder 프로세스가 USock 시그널리스트에 명시된 시그널을 하위 계층에게 전달하는 경로
TRsapR_out	시스템 명세의 TRsapS 채널과 연결
	Responder 프로세스가 TRout 시그널리스트에 명시된 서비스 프리미티브 시그널을 상위 계층(WTP 사용자)에게 전달하는 경로

표 3 내부 시그널 라우트

시그널 라우트	설 명
internal_cli	클라이언트 관리 프로세스와 트랜잭션 관리 프로세스간의 통신.
	Client_Manger 프로세스가 USock, TRin 시그널리스트에 명시된 PDU 시그널과 서비스 프리미티브 시그널을 TR_Manger 프로세스에 전달하는 경로.
internal_tr	클라이언트 관리 프로세스와 트랜잭션 관리 프로세스간의 통신.
	TR_Manger 프로세스가 Cctrl 시그널을 Client_Manger 프로세스에 전달하는 경로.
internal_ini	트랜잭션 관리 프로세스와 요구자 프로세스간의 통신.
	TR_Manger 프로세스가 USock, TRin 시그널리스트에 명시된 PDU 시그널과 서비스 프리미티브 시그널을 Initiator 프로세스에 전달하는 경로. Initiator 프로세스가 Ictrl 시그널을 TR_Manger 프로세스에 전달하는 경로.
internal_res	트랜잭션 관리 프로세스와 응답자 프로세스간의 통신.
	TR_Manger 프로세스가 USock, TRin 시그널리스트에 명시된 PDU 시그널과 서비스 프리미티브 시그널을 Responder 프로세스에 전달하는 경로. Responder 프로세스가 Rctrl 시그널과 Lastid_ret 시그널을 TR_Manger 프로세스에 전달하는 경로.

```
SYNTYPE Index = INTEGER
  constants 1:10
ENDSYNTYPE;

NEWTYPE MSET_PR Powerset(Index)
ENDNEWTYPE;

NEWTYPE MTable_Element Struct
  h HANDLE;
  tid TID_TYPE;
  class Integer;
  pid PID;
ENDNEWTYPE;

NEWTYPE MTABLE_I Array(Index, MTable_Element)
ENDNEWTYPE;

NEWTYPE MTABLE_R Array(Index, MTable_Element)
ENDNEWTYPE;
```

그림 1 매핑 테이블의 구조 정의

```
void xInEnv(...)
{
  FD_SET(/* 대표소켓 */, ...);
  FD_SET(/* IPC소켓 */, ...);
  if ( select (...) > 0)
    if FD_ISSET(/* 대표소켓 */, ... ) {
      recvfrom(/* 대표소켓 */, buff, ...);
      switch (/* 타입 */) {
        case Invoke PDU:
          xGetSignal(/* Invoke PDU */ , , );
          /* 디코딩 */
          SDL_Output( , , );
        case Result PDU:
            생략 -
      }
    }
    if FD_ISSET(/* IPC소켓 */, ...) {
      recv(/* IPC소켓 */, buff, ...);
      switch (/* 타입 */) {
        case TRInvoke_req:
          - 생략 -
      }
    }
}
```

그림 2 환경함수 xInEnv 구조

표 4 WTP 응용 프로그래밍 인터페이스 함수

프리미티브	함 수	설 명
	serverWTP	서버에서 WTP 프로토콜 프로세스의 생성
	clientWTP	클라이언트에서 WTP 프로토콜 프로세스의 생성
	acceptWTP	WTP 프로토콜 프로세스로부터의 IPC 채널 요청의 수락
	closeWTP	WTP 프로토콜 프로세스의 종료
TRInvoke_req	write_invoke	새로운 트랜잭션을 초기화
TRInvoke_ind TRAbort_ind	read_invoke	Invoke 메시지의 수신
TRResult_req	write_result	Result 메시지의 송신
TRResult_ind TRAbort_ind	read_result	Result 메시지의 수신
TRInvoke_res TRResult_res	write_ack	수신한 메시지에 대한 응답으로 Ack 메시지의 송신
TRInvoke_con TRResult_con TRAbort_ind	read_ack	Ack 메시지의 수신
TRAbort_req	write_abort	수행중인 트랜잭션의 취소

표 5 서버 PICS (M:Mandatory, O:Optional)

Identifier	WTP function	type	
WTP-SC22	Transaction Class 0	Initiator	M
WTP-SC23		Responder	M
WTP-SC24	Transaction Class 1	Initiator	M
WTP-SC25		Responder	M
WTP-SC26	Transaction Class 2	Initiator	O
WTP-SC27		Responder	M
WTP-SC28	User acknowledgement		M
WTP-SC30	Separation		M
WTP-SC31	Re-transmission until acknowledgement		M
WTP-SC32	Transaction abort		M
WTP-SC33	Error Handling		M
WTP-SC35	Asynchronous transactions		O
WTP-SC36	Transaction identifier verification	Initiator	M
WTP-SC37		Responder	O

```
void xOutEnv(...)
{
  switch (/* 타입 */) {
    case Invoke PDU:
      /* 인코딩 */
      sendto(/* 대표소켓 */, buff, ...);
      xReleaseSignal();
    case Result PDU:
      - 생략 -
  switch (/* 타입 */) {
    case TRInvoke_req:
      - 생략
  }
}
```

그림 3 환경함수 xOutEnv 구조

이 해 동

1999년 2월 경북대학교 컴퓨터과학과(이학사). 2001년 2월 경북대학교 컴퓨터과학과(이학석사). 2001년 2월 ~ 현재 한국전자통신연구원 연구원. 관심분야는 AAA (Authentication, Authorization and Accounting), 정보보호, 이동 컴퓨팅, 프로토콜 공학

정 호 원

2000년 경북대학교 통계학과(이학사). 2000년 ~ 현재 경북대학교 컴퓨터과학과 석사과정. 관심분야는 프로토콜 공학, 컴퓨터통신, 홈 네트워킹

원 유 재

1985년 2월 충남대학교 계산통계학과(이학사). 1987년 2월 충남대학교 전산학(이학석사). 1998년 8월 충남대학교 전산학(이학박사). 1987년 2월 ~ 2001년 2월 한국전자통신연구원 책임연구원, 무선인터넷정보보호연구팀장. 2001년 3월 ~ 현재 안철수연구소, IA 시큐리티 기술이사. 관심분야는 정보보호, 프로토콜 공학, 이동컴퓨팅

임 경 식

1982년 경북대학교 전자공학과(공학사). 1985년 한국과학기술원 전산학과(공학석사). University of Florida 전산학과(공학박사). 1985년 ~ 1998년 한국전자통신연구원 책임연구원, 실장. 1998년 ~ 현재 경북대학교 컴퓨터과학과 조교수. 관심분야는 이동 컴퓨팅, 무선 인터넷, 홈 네트워킹, 컴퓨터통신