차세대지능망 응용 프로토콜 분석/설계 및 검증

도 현 숙†

요 약

차세대 지능망에서는 서비스와 하부 통신망에 투명한 범용 지능망 응용 프로토콜(INAP)을 지향하고 있다. 본 논문에서는 차세대 지능망에 적합한 표준화된 INAP 구조를 도입하여, 이에 추가적으로 기능 요소들을 정의함으로써 INAP설계구조를 본 논문에서 제안하였다. 또한 객체지향 방법론을 적용하여 IN CS-1의 목표 서비스 중의 하나인 AAB서비스를 대상으로 INAP을 분석하고 설계하였다. 분석 단계에서는 ObjectGEODE에서 제공하고 있는 변형된 형태의 OMT방법을 적용하였으며, 설계 단계에서는 SDL을 사용하여 설계하였다. 설계된 시스템을 구현하는 것에 앞서 이 시스템이 제대로 설계되었는가를 검증하였으며 그에 대한 과정과 결과를 보이고 있다.

A Design and Analysis, Simulation of Intelligent Network Application Protocol

Hyunsook Do†

ABSTRACT

The AIN(Advanced Intelligent Network) employs the Intelligent Network Application Protocol(INAP) which is transparent to services and low layer telecommunication networks. In this paper, we introduce the standard structure of INAP, which has flexible structure for easy expansion as AIN evolves, and we propose a structure of INAP by defining its functional elements. Also we design, analyze the INAP for AAB(Automatic Alternative Billing) service, which is one of the target services of IN CS-1 using object-oriented methodology. We use OMT method which is modified by ObjectGEODE in the analysis phase and SDL language for design. Furthermore we simulate the INAP system to prove that the system has been correctly built and supplies the service correctly for which it was designed.

1. 서 론

1.1 목 적

초기 지능망에서는 서비스 제어 기능과 호 처리 기능이 분리되어 있지 않아 새로운 서비스의 추가나 수정이 어려웠다. 또한 물리적 장치간의 프로토콜이 제공할 서비스에 종속되어 새로운 서비스를 도입할 때 관련된 소프트웨어의 추가 개발이 요구되므로 다양한 서비스 도입에 한계가 있고 전화망만을 대상으로 하는 단점이 있었다.

이러한 단점을 극복하기 위해서 국제 표준화 기구인 ITU-T(International Telecommunication Union Telecommunications Standardization Sector)에서는 재사

† 정 회 원:한국전자통신연구원 지능망구조연구실
논문접수:1996년 7월 18일, 심사완료:1997년 5월 2일

용 가능한 모듈 구조를 가진 차세대 지능망에 대한 연구를 활발히 하고 있다. 차세대 지능망에서는 서비스나 망에 독립적인 망 기능 구조를 정의하며, 이러한 기능들은 물리적으로 분산되어 망 운용자가 망의 특정 기능을 수행하는 플랫폼들을 사용하여 전체망을 구축하는데 융통성을 제공한다[6].

기능의 분산은 결국 이들간의 통신을 필요로 한다. 즉, 망 기능들은 서비스를 제공하기 위해 상호 신호를 위한 프로토콜을 사용한다. 차세대 지능망에서는 서비스와 하부 통신망에 투명한 범용 지능망 응용 프로토콜(INAP)을 지향하고 있어 전화망과 ISDN, B-ISDN 및 FPLMTS를 기반망으로 다양한 서비스를 동시에 수용할 수 있는 통합 플랫폼의 구축이 기대되고 있다.

본 논문에서는 차세대 지능망이 CS (Capability Set)-2/3로 진화함에 따라 프로토콜도 이를 수용할 수 있는 확장 구조를 가져야 한다는 필요성에 부합하기 위해 차세대 지능망에 적합한 INAP 구조를 도입하여, 이에 추가적으로 기능 요소들을 정의함으로써 INAP 설계구조를 제안하였다. 또한 IN CS-1의 목표 서비스 중의 하나인 AAB (Automatic Alternate Billing) 서비스를 대상으로 INAP을 확장성, 유지보수, 재사용성이 뛰어난 객체지향 방법론에 적용하여 분석하고 설계하였으며, 구현에 앞서 시스템이 제대로 설계되었는가를 검증하기 위해 모의 테스트를 수행하였다.

1.2 배 경

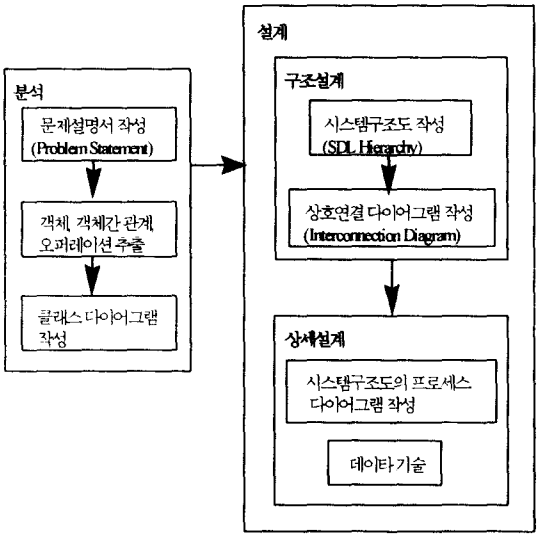
차세대 지능망은 기능적인 관점에서 볼 때 다음의 기능요소들로 구성된다.

교환기의 기본호처리로부터 지능망 서비스를 감지하여 서비스 수행을 요구하는 서비스 교환 기능(SSF: Service Switching Function), 요청된 지능망 서비스를 수행하기 위한 서비스 제어 기능(SCF:Service Control Function), 서비스 수행 중 사용자 인터페이스를 위한 특수 자원 기능(SRF:Specialized Resource Function), 서비스 수행에 필요한 데이터를 관리하기 위한 서비스 데이터 기능(SDF:Service Data Function)들로 구성되어 있다.

INAP은 지능망의 기능 실체들간의 상호 동작에 필요한 신호를 지원하기 위한 응용 프로토콜이며, IN CS-1 INAP은 SCF-SSF, SCF-SRF 그리고 SCF-SDF 기능 실체들간의 응용 계층 인터페이스를 지원한다. 본 논문에서는 서비스 제어 기능 내에 서비스 데이터를 포함하고 있는 형태를 가정하고 있으므로 SCF-SDF간의 인터페이스는 고려하지 않는다.

객체지향 INAP을 분석, 설계, 검증하는 단계에서 객체지향 도구인 ObjectGEODE를 사용하였다. ObjectGEODE에서는 객체지향 기법 중에서 Rumbaugh가 창안한 OMT (Object Modeling Technique)방법의 변형된 형태와 ITU-T에서 통신 소프트웨어의 설계 및 자동 생성에 적합한 언어로 권고하고 있는 SDL (Specification & Description Language) 언어를 결합시켜 분석, 설계하는 방법을 지원하고 있으며, 설계된 시스템을 테스트해 볼 수 있는 시뮬레이터를 제공함으로써 검증을 가능하게 한다[9], [10], [11], [12].

(그림 1)에서 객체 지향 방법론을 적용한 분석 및 설계 과정을 보이고 있다. 앞서 언급된 바와 같이 분석 단계에서는 ObjectGEODE에서 제공하고 있는 변형된 형태의 OMT방법을 적용하여 문제 설명서를 작성하여 클래스를 추출하고 클래스들의 상관 관계 등을 추출하여 클래스 다이어그램을 작성한다. 설계 단계에서는 OMT방법에서 제안하고 있는 Event Trace Diagram (ETD)이나 State Transition Diagram(STD)으로 기술하지 않고 SDL표기를 이용하여 시스템을 설계하였는데, 이는 SDL이 ETD나 STD보다 상태 천



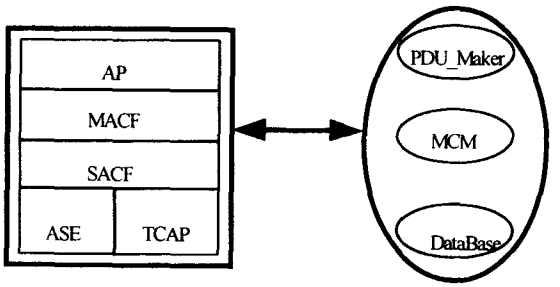
(그림 1) 객체지향론에 의한 분석, 설계 절차
(Fig. 1) Analysis and design by object oriented method.

이를 기술하는데 있어 형식적(formal)인 방법을 제공하기 때문이다.

설계된 시스템이 완전한 코드로 생성되기 이전에 시뮬레이터를 통해 시스템이 에러 없이 설계가 되었는가와 요구 사항에 맞게 설계되었는가를 시험해 볼 수 있으며, 반복적인 테스트, 에러 수정 작업이 이루어진다. 이 단계 처리가 끝나면, 설계 단계에서 SDL로 표현되었던 시스템은 코드 생성기를 이용하여 C Code로 변환되고, 컴파일러를 통해 실행 가능해진다.

2. INAP구조

(그림 2)는 본 논문에서 제안하고 있는 INAP의 설계구조이다. 이 INAP설계구조는 객체지향 개념을 바탕으로 하며, 이들 구성 요소 하나 하나가 객체로 정의될 수 있다. 구현을 위한 INAP 설계구조는 ITU-T에서 제안한 INAP구조와 본 논문에서 추가로 정의한 기능요소들로 구성되어 있다. 이는 INAP기능을 분석을 하면서 INAP 각 구성요소에 공통적으로 필요한 필수 기능들과 기존 INAP에 정의되어 있지 않은 부분을 정의한 것이다.



(그림 2) INAP설계구조
(Fig. 2) Proposed design structure of INAP

ITU-T에서 제안한 INAP구조의 구성요소는 MACF (Multiple Association Control Function), SACF(Single Association Control Function), ASE(Application Service Element), TCAP(Transaction Capabilities Application Part)으로 이루어져 있으며[1], 추가로 정의된 기능요소들은 PDU_Maker(ProtocolDataUnit_Maker), MCM(Multiple Call Manager), DataBase로 이루어져 있다. 이들 기능은 다음과 같다.

- MACF는 한 기능실체에 하나 이상의 SACF가 존재할 경우 다중 응용결합의 제어 기능을 제공한다.
- SACF는 기능실체간에 단일접속 상호작용이 일어날 때 AP가 ASE를 사용하는 기능을 담당한다.
- ASE는 하나 이상의 오퍼레이션으로 구성된다. 원격 오퍼레이션을 제공하기 위한 기능 모듈로서 오퍼레이션의 호출자와 수행자를 규정한다.
- TCAP은 두 사용자간 트랜잭션을 통하여 원격 오퍼레이션과 이에 대한 응답을 송수신할 수 있는 서비스를 제공한다.
- MCM은 다중호를 관리할 수 있는 관리자로서 이를 통해 다중호 처리를 할 수 있는 메카니즘을 제공할 수 있도록 하였다. 이 기능은 INAP규격에 정의되어 있지 않은 부분으로 다중호 처리 기능은 INAP구현에 있어 필수적인 기능이기 때문에 새로 정의한 모듈이다.
- PDU_Maker는 구성요소들간에 주고 받는 데이타 형식을 변환시켜주는 기능을 해준다. 이 기능은 INAP 구성요소들에게 공통적으로 필요한 기능이며 그 기능을 한번만 정의하여 사용함으로써 중복성을 제거할 수가 있다.
- DataBase는 MACF와 ASE, MCM이 사용하는 데이타를 관리해 주는 역할을 한다.

3. INAP의 객체지향 분석

OMT방법에 따른 분석단계에서는 먼저 구현하고자 하는 시스템이 무슨 일을 하는가에 대한 문제 설명서(Problem Statement)를 작성해야 하며 기술된 문제와 사람이 가지고 있는 지식을 토대로 클래스와 클래스간의 관계, 클래스가 하는 일을 추출하여 클래스 다이어그램을 작성해 나간다[8].

3.1 문제 설명서(Problem Statement)

다음은 INAP의 기능요소를 중심으로 INAP이 하는 일에 관하여 서술적으로 기술하였다. 이는 분석을 위한 가장 기초적인 작업으로서, 반복적으로 수정, 추가등을 통하여 만들어진다. 여기서는 서비스 개시를 위한 오퍼레이션 처리 절차만을 보였다.

- INAP은 SCF, SSF, SRF로부터 메시지를 송신하고

자 하는 실체(SCF, SSF, SRF 중 하나)에게 송신해 줄 것을 요구 받는다. INAP은 요구를 받고 전송할 메시지가 문법에 맞는지를 확인하고 메시지를 구성하여 수신 대상이 되는 실체에게 메시지를 전송한다. INAP은 여러 서비스, 다중 호를 처리해주기 때문에 송수신되는 메시지 순서를 제어해 준다.

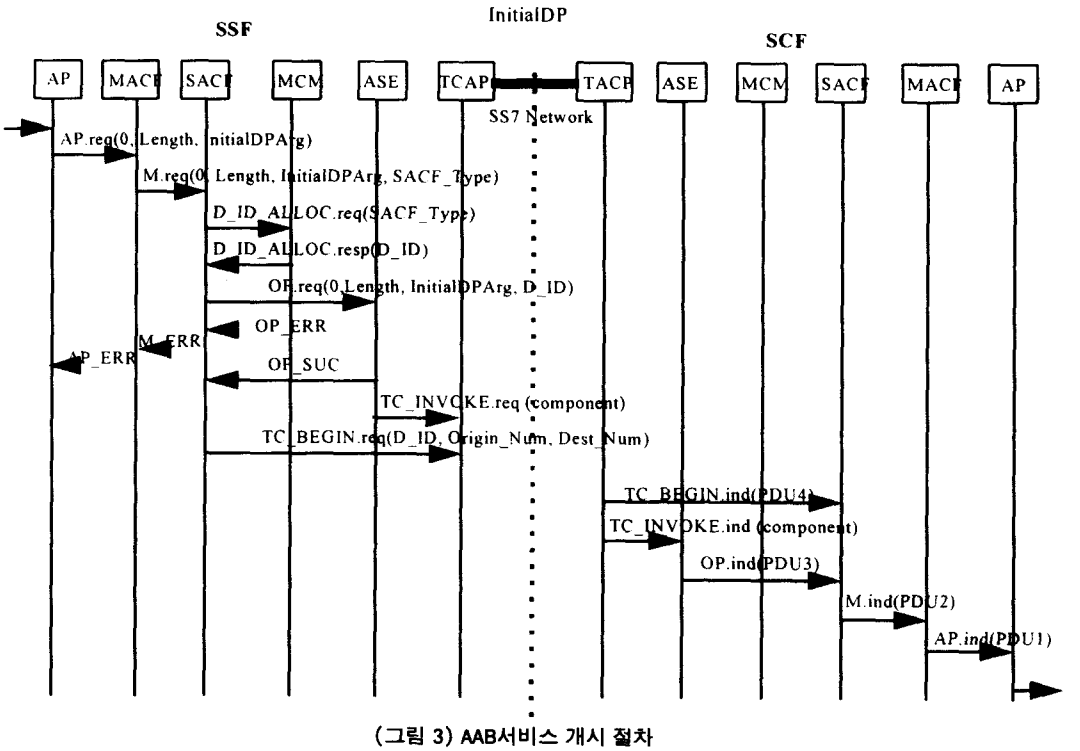
- 지능망서비스가 요청되면 MACF는 AP로부터 AP.req 프리미티브를 통하여 메시지를 수신한다. MACF는 SACF_Type을 결정하고 M.req 프리미티브를 통하여 SACF로 메시지를 송신한다.

SACF는 서비스 개시 오퍼레이션을 받으면 D_ID_ALLOC.req프리미티브를 통해 MCM에게 D_ID(Dialog_IDentifier) 할당을 요구한다. D_ID는 새로운 서비스 요구 발생할 때마다 할당이 이루어진다. MCM은 SACF로부터 D_ID_ALLOC.req를 통해 수신한 SACF_Type에 대해 현재 미사용 중인 D_ID를 할당하고 D_ID_ALLOC.resp 프리미티브를 통하여 SACF에게 D_ID를 통보한다. SACF는 D_ID를 할당받은 뒤에 OP.req를 통하여 ASE에게 메시지를 송신한다. 이때 SACF가 개시 오퍼레이션 외의 오퍼레이션을 수신하면 이미 할당된 D_ID를 사용한다.

ASE는 수신한 메시지 내용을 이미 정의된 ASE의 ASN.1 표기법과 문법적으로 비교 검사하여 에러를 검사한다. 만약 에러가 있으면 OP_Err 프리미티브를 사용하여 SACF로 에러상황을 통보하고, 없으면 OP_Suc 프리미티브를 통하여 이를 SACF에게 통보한다.

SACF는 에러를 수신하면 M_Err 프리미티브를 통하여 MACF에게 에러상황을 통보하고 MACF는 다시 AP_Err 프리미티브를 통하여 AP에게 이를 통보한다. 만약 에러가 없으면 ASE는 TC_INVOKE.req 프리미티브를 통하여, SACF는 TC_BEGIN.req 프리미티브를 통하여 TCAP에게 메시지를 전송한다.

TCAP은 ASE와 SACF로부터 메시지를 수신하면 이를 TSL(Transaction Sub-Layer)을 통하여 상대통신실체로 송신한다. 상대통신실체로부터 메시지를



(그림 3) AAB서비스 개시 절차
(Fig. 3) AAB service initiating procedure

받는 경우는 위에서 기술한 절차를 반대로 수행을 한다.

(그림 3)은 문제 설명서에서 설명한 서비스 개시 절차, 즉 초기감지(InitialDP)처리 절차를 INAP클래스들 사이에서 송수신되는 메시지 흐름으로 보였다.

3.2 클래스와 오퍼레이션, 관계(association) 추출

OMT방법에 의한 클래스와 클래스의 오퍼레이션, 클래스간의 관계 추출 방법은 문제 설명서(Problem Statement)를 중심으로 이루어지고, 추가적으로 사람이 가지고 있는 지식(Domain Knowledge), 실제 경험(Real-World Experience)을 반영한다. 먼저 클래스를 추출하기 위해서는 문제 설명서에서 명사를 찾아 후보 클래스를 추출하여, 그 중에서 부적절하다고 여겨지는 클래스를 삭제한 뒤에 최종 클래스 리스트를 작성한다. 다음은 찾아낸 클래스에 대한 간단한 설명을 담고 있는 자료 사전을 구성하는 것인데 여기서는 클래스가 하는 일, 즉 오퍼레이션을 찾아 오퍼레이션 리스트를 작성하였다.

한 클래스간의 관계는 문제 설명서의 동사구에서 찾아 정의를 하고, 클래스에 필요한 애트리뷰트의 정의는 문제 설명서의 소유격 표현, 혹은 적절하다고 여겨지는 명사를 추출함으로써 이루어진다. 이런 방법들을 이용하여 본 논문에서는 다음과 같은 클래스, 오퍼레이션, 클래스간의 관계, 클래스의 애트리뷰트를 정의하였다.

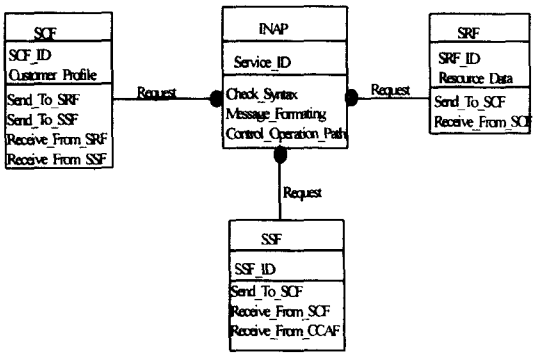
- 클래스(object): INAP, SCF, SSF, SRF, MACF, SACF, ASE, MCM, TCAP, AP-Primitive, M_Primitive, D_ID_Primitive, OP_Primitive, TC_Transaction, TC_Component, DataBase
- 오퍼레이션(operation): 메시지 순서를 제어한다. SACF_Type을 결정한다, PDU2를 생성한다, D_ID할당을 요구한다, D_ID해제를 요구한다, PDU3을 생성한다, PDU4를 생성한다. D_ID를 할당한다, D_ID를 해제한다, 문법을 검사한다, component를 만든다, 다이얼로그를 설정한다, 메시지를 구성한다.
- 관계(association): 통보한다. 전달한다. 수신한다. 요구한다.
- 애트리뷰트(attribute): Service_ID, SCF_ID, SSF_ID, SRF_ID, Customer_Profile, Resource_Data, SACF_Type, OP_Code

3.3 클래스 다이어그램

(그림 4)와 (그림 5)는 3.2절에서 추출한 클래스와 클래스의 오퍼레이션, 클래스간의 관계, 클래스의 애트리뷰트를 클래스 다이어그램으로 나타낸 것이다.

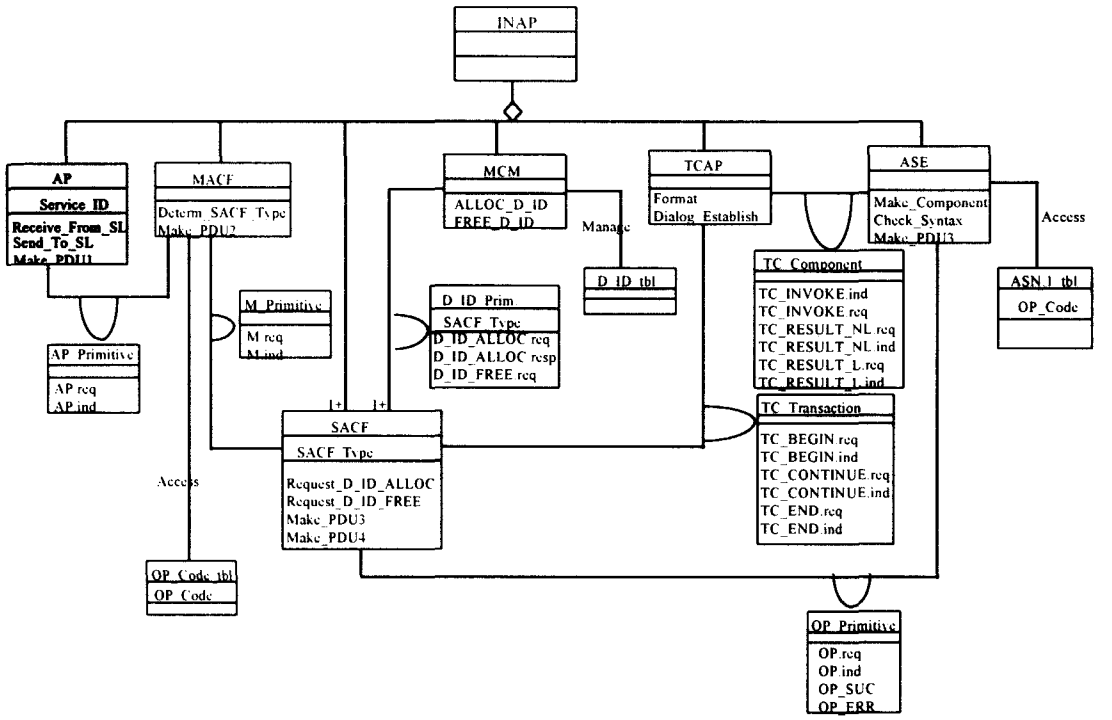
여기서 클래스 다이어그램의 표기법은 OMT방법을 따른 것으로, 사각형 박스 안에 세 영역으로 구분되어 있는데, 위에서 차례로 클래스의 이름, 클래스의 애트리뷰트, 클래스의 오퍼레이션을 나타낸다. 클래스와 클래스간에 그려진 선은 서로 관계가 있음을 나타내며, 선 위에 표기된 내용은 서로 어떤 관계를 갖고 있는지를 나타낸다. 예를 들어 SCF와 INAP간에는 요구(Request)가 명기되어 있는데 이는 SCF가 INAP에게 어떤 일을 해줄 것을 요구한다는 의미이다. 이는 단방향만 표현할 수도 있고 쌍방향 모두 표현할 수 있다.

(그림 4)는 INAP시스템과 그 주변환경과의 관계를 나타낸 것으로 이는 시스템을 이해하는데 도움을 주며, 본 논문에서는 SCF-SDF간의 인터페이스는 고려하지 않으므로 INAP이 인터페이스를 갖는 기능실체들은 SCF, SSF, SRF이다.



(그림 4) 클래스 다이어그램1
(Fig. 4) Class diagram1

(그림 5)는 INAP시스템의 클래스 다이어그램이다. 여기서 INAP은 모두 6개의 클래스로 구성되어 있고, 클래스들이 사용하는 데이타 베이스들도 클래스로 정의하였다. 두개의 클래스들간에 공통적으로 관계



(그림 5) 클래스 다이어그램2
(Fig. 5) Class diagram2

(association)가 있는 것들도, 즉 여기서는 프리미티브들인데, 하나의 클래스로 정의하였다.

4. INAP의 객체지향 설계

설계 단계에서는 국제표준화 단체인 ITU-T에서 통신 소프트웨어의 설계 및 자동 생성에 적합한 언어로 권고하고 있는 SDL을 이용하였다. 설계 단계는 구조설계와 상세설계절차로 나뉘는데 구조설계에서는 분석단계에서 만든 클래스 다이어그램을 바탕으로 하여 구현될 시스템의 전체적인 윤곽을 잡아 시스템 구조도를 작성하고, 시스템 구성요소들간에 주고 받는 메시지, 구성요소들의 오퍼레이션들로 상호연결 다이어그램(Interconnection Diagram)을 작성한다. 상세설계에서는 구조설계에서 작성한 시스템 구조도의 최하위 레벨에 있는 프로세스들의 유한상태머신(Finite State Machine)을 SDL로 기술한다[9]. 이를 이하 프로세스 다이어그램이라 한다. 또한 각 프로세스들이 사용하는 구체적인 데이타들도 정의한다. 설계 단계중 ASE를 상세 설계하는 부분에서는 IN CS-1서비스의 하나인 AAB서비스를 대상으로 하여 프로세스 다이어그램을 작성하였으며, 이는 확장 가능하다.

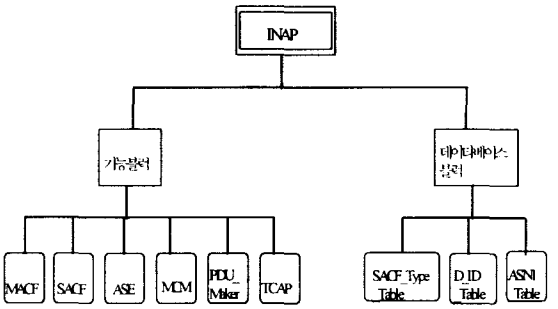
4.1 구조설계

구조 설계는 구현하고자 하는 시스템의 정적인 측면과 동적인 측면을 시스템 구조도와 상호연결 다이어그램을 이용하여 기술한다. 시스템 구조도는 시스템을 구성하는 컴포넌트(system, block, process, procedure, service, data declaration)들의 계층적인 구조를 보여 주며, 시스템의 정적 표현을 보여 준다.

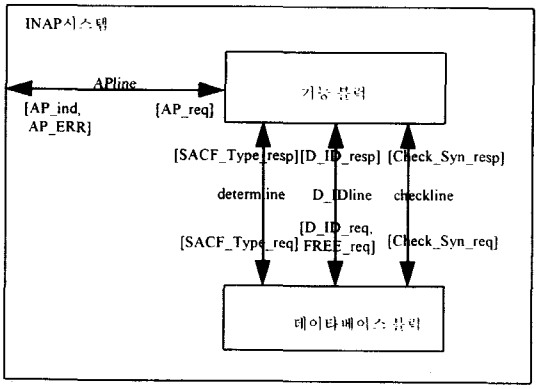
상호연결 다이어그램은 시스템 구조도의 각 레벨을 구성하는 컴포넌트들간과 외부환경과 주고 받는 메세지의 흐름을 보여주며, 시스템의 정적인 측면과 동적인 측면을 함께 보여 준다. 상호연결 다이어그램에서 정의하는 메시지와 컴포넌트들간의 연결 관계는 상세설계시 프로세들이 주고 받는 시그널 등을 정의할 때 필요한 요소들이다.

(그림 6)은 INAP시스템 구조도를 나타낸다. 이 시

스템 구조도는 크게 INAP의 기능실체들로 이루어진 기능요소 블럭과 이들이 사용하는 데이타 베이스 블럭으로 나뉘어 진다. 기능실체 블럭은 MACF, SACF, ASE, MCM, PDU_Maker, TCAP 프로세스들로 구성되며, 데이타 베이스 블럭은 SACF_Type_Table, D_ID_Table, ASN1_Table로 구성되어 있다. (그림 7)은 상호연결 다이어그램 중 블럭들간, 외부 환경과의 통신 흐름을 나타낸 것이다.



(그림 6) 시스템 구조도
(Fig. 6) System hierarchy



(그림 7) 상호연결 다이어그램
(Fig. 7) Interconnection diagram

기능 블럭은 데이타 베이스 블럭과 세 개의 채널, determline, D_IDline, checkline을 통해 각각 정의된 시그널을 주고 받으며, 외부 환경과는 Apline이라는 채널을 통해 시그널을 주고 받는다. 각 블럭을 한 단계씩 더 들어가면 프로세스들간에 주고 받는 시그널들이 정의 되어 있으며 이때 외부 환경은 그들이 속해 있지 않은 블럭들이 된다. 상호연결 다이어그램에서 정의된 시그널들은 상세설계에서 프로세스 다이어그램 작성시 반영이 된다.

4.2 상세설계

상세설계에서는 앞서 설명한 바와 같이 구조설계에서 작성한 시스템구조도의 최하위 레벨에 있는 프로세스들을 SDL로 기술하는데, 여기에서는 주로 구현 관점으로 코드화 할 수 있는 정도로까지 상세하게 작성해 나간다.

본 논문에서는 IN CS-1 서비스 중의 하나인 AAB 서비스를 처리할 수 있는 ASE를 설계하였으며, ASE를 제외한 요소들은 모든 서비스에 범용적으로 사용할 수 있도록 설계하였다. 이는 ASE를 확장함으로써 지능망 서비스들을 수용할 수 있음을 의미한다.

4.2.1 AAB서비스 수행 절차

AAB서비스(자동대체과금 서비스)는 사용자에게 다른 전화기로부터 전화가 가능하도록 하며 이 서비스에 대한 과금은 서비스에 규정되어 있는 사용자 계좌에 부가하는 서비스이다. (그림 8)은 AAB 서비스를 기능실체들간에 주고받는 정보흐름으로 나타낸 것이다[2].

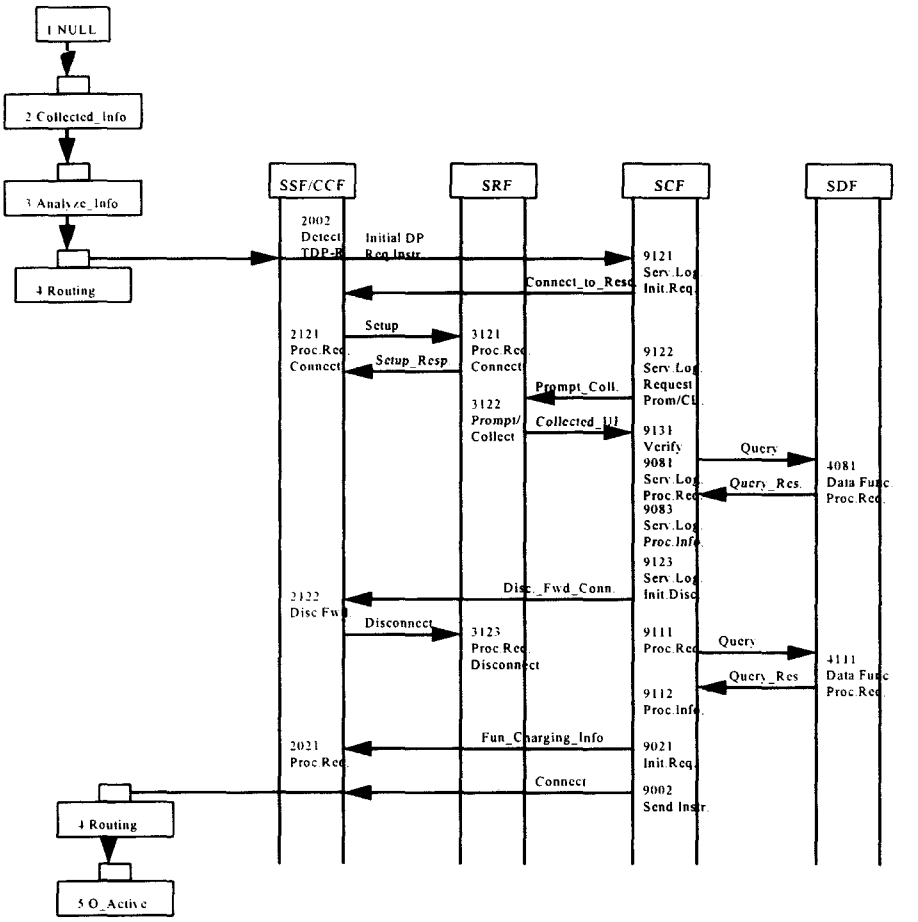
(그림 8)에 대한 설명은 다음과 같다.

- AAB서비스 사용자는 자신의 AAB 서비스번호를 이용하여 자신의 전화기 또는 공중전화기로부터 SSF로 AAB서비스 요청을 한다.
- SSF는 자신에게 연결된 로컬 교환기(LE) 또는 자신에게 연결되어 있는 단말기로부터 AAB서비스 요청을 받아 AAB서비스 요청임을 판단하고 초기감지(InitialDP) 정보흐름을 이용하여 SCF에게 AAB 서비스를 요구한다.
- SCF는 서비스 번호를 해석하여 가입자번호에 해당하는 비밀번호가 필요함을 인지하고, 자원연결(Connect_to_Resource) 정보흐름을 통해 SSF에게 SRF와 연결을 요구한다.
- SRF의 연결요구를 받은 SSF는 SRF와 연결된다.
- SCF는 사용자정보수집(Prompt&Collect UserInformation) 정보흐름을 이용하여 비밀번호를 입력할 것을 SRF에게 요구한다.
- SRF를 통해 입력된 가입자의 비밀번호가 수집정보(CollectedUserInformation) 정보흐름을 통해 SCF

로 전달된다.

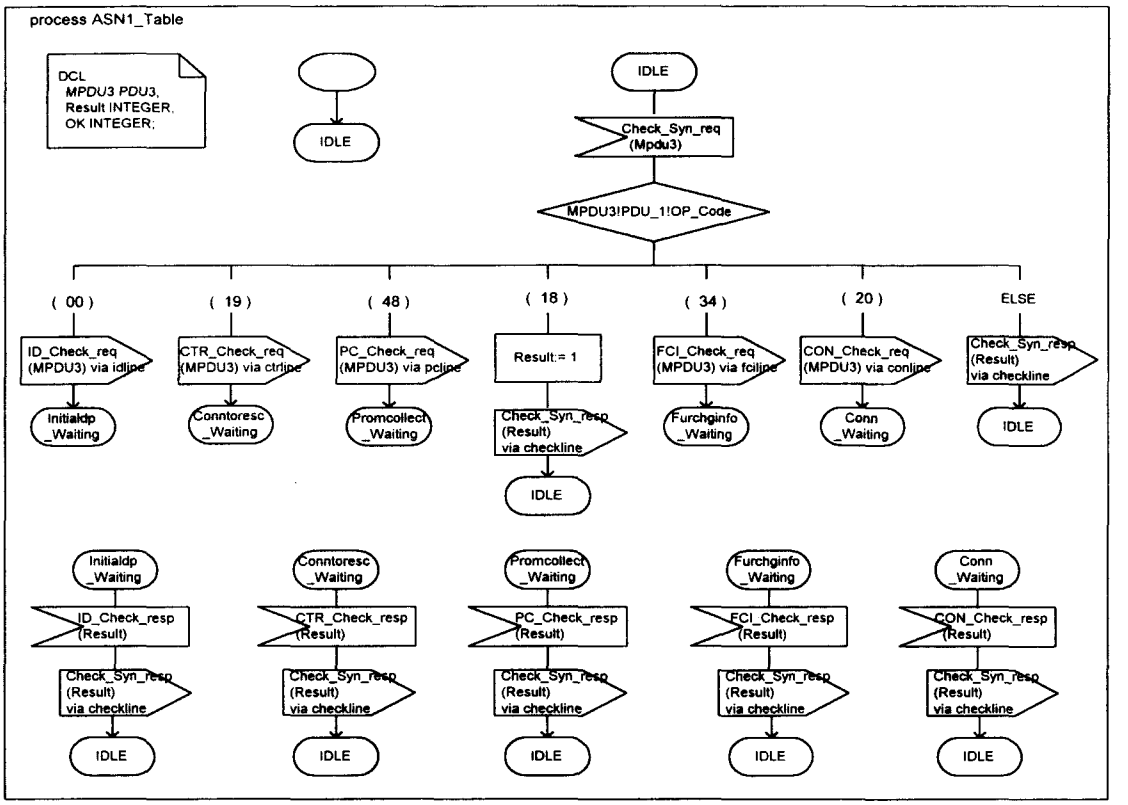
- 가입자가 입력한 비밀번호가 일치하면 SCF는 SSF에게 SRF와의 접속을 해제할 것을 순방향연결해제(Disconnect ForwardConnection) 정보흐름을 이용하여 요구한다.
- SSF는 SRF와의 접속을 해제한다.
- SCF는 호에 대한 과금이 AAB번호로 과금되도록 과금정보제공(Furnish ChargingInformation) 정보흐름을 이용하여 SSF로 전달하고 이용자가 요구한 착신번호로 착신되도록 연결(Connect) 정보흐름을 통해 전달한다.
- SSF는 SCF의 명령에 의해 AAB서비스 이용자 호를 착신측과 연결하며 과금이 SCF의 요구대로 AAB번호에 과금되도록 처리한다.
- 비밀번호가 불일치할 경우 SCF는 SSF에게 적절한 안내방송을 할 것을 명령한다.
- SSF는 SCF로부터의 안내방송요청을 받고 SRF를 통해 안내방송을 이용자에게 보낸다.

위와 같이 기술된 AAB서비스를 처리할 수 있는 ASN1_Table프로세스를 (그림 9)에 나타내었다. ASN1_Table을 제외한 나머지 프로세스들은 모든 서비스에 공통적으로 사용될 수 있기 때문에 여기서는 AAB서비스에 관련된 ASN1_Table만을 보였다.



(그림 8) AAB서비스 시나리오

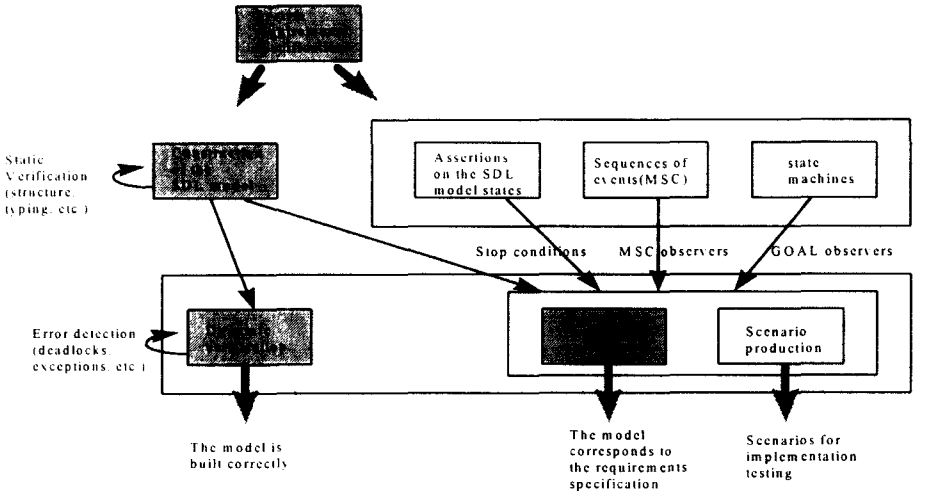
(Fig. 8) AAB service scenario



(그림 9) ASN1_Table 프로세스 다이어그램
(Fig. 9) ASN1_Table process diagram

5. 모의 테스트(Simulation)

설계한 시스템이 에러 없이 제대로 동작하는가와 요구 사항에 맞게 동작하는가를 확인하기 위해서 ObjectGEODE에서 제공하는 도구들 중에 하나인 시뮬레이터를 사용하여 검증하였는데, ObjectGEODE에서 제공하는 모든 도구들에 대한 기능은 (그림 10)에 나타나 있으며, 빗금으로 표시된 부분이 검증을 위해 주로 연구된 부분이다. 본 논문에서는 (그림 10)에서 설명하고 있는 절차의 일부, 즉 빗금으로 표시된 부분을 수행하였는데, 우선 시스템 요구 사항을 분석 단계에서 분석한 뒤 설계 단계에서 SDL 다이어그램을 작성하여 SDL 문법이나 타이핑 에러 등을 검출하는 정적인 Verification을 거친다. 정적인 Verification이 끝나면 시스템의 논리적인 에러를 검출할 수 있는 동적인 Verification과 요구 사항에 맞게 동작하는 가를 확인할 수 있는 Validation 과정을 거침으로써 검증 절차를 끝낸다. ObjectGEODE에서 지원 가능한 검증 방법에는 사용자와 상호작용을 가지면서 테스트를 하는 방법, 시뮬레이터에서 무작위로 검증 순서를 정하여 하는 방법, 가능한 모든 경우를 적용하는 방법이 있는데, 본 논문에서는 하나의 서비스, 즉 AAB서비스를 대상으로 사용자와 상호작용을 갖는 방법과 가능한 모든 경우를 적용하는 방법으로 검증 하였다. ObjectGEODE에서 제공하고 있는 사용자 상호작용 시뮬레이터는 시스템 오류를 찾아 내는데 많은 도움을 주었으며, 에러가 발견되는 부분을 SDL로 보여줌으로써 에러 수정을 쉽고 신속하게 할 수 있었다. 시뮬레이션은 Sun Server 1000E, Solaris4.2x 환경에서 이루어졌으며, AAB서비스 시나리오 절차를 입력 데이터로 하였다. (그림 11)에서는 사용자와 상호작용을 갖는 방법의 절차를 보이고 있는데, InitialDp와 ConnectTo Resource오퍼레이션 송수신 절차를 수행하는 과정을 단계별로 보이고 있고, 여기서



(그림 10) 모의 테스트 방법

(Fig. 10) The method of simulation

```
> feed apline ap_req(0,10,3,10,10,10,1,1,0,'8612583','8604894')
> feed tc_line tc_continue2_req(19,10,3,0,0,0,0,0,0,'8612583','8604894',1)
> track trans macf(1) : start
> fire macf(1) : start
step = 1
trans macf(1) : start
> track trans macf(1) : start
> track trans macf(1) : from_wait_input_input_ap_req with ap_req(0 , 10 , 3 , 10 , 10 , 10 , 1 , 1 , 0 , '8612583' , '8604894')
> track trans macf(1) : from_wait_input_input_ap_req with ap_req(0 , 10 , 3 , 10 , 10 , 10 , 1 , 1 , 0 , '8612583' , '8604894')
> fire macf(1) : from_wait_input_input_ap_req with ap_req(0 , 10 , 3 , 10 , 10 , 10 , 1 , 1 , 0 , '8612583' , '8604894')
step = 2
trans macf(1) : from_wait_input_input_ap_req with ap_req(0 , 10 , 3 , 10 , 10 , 10 , 1 , 1 , 0 , '8612583' , '8604894')
input ap_req from env to macf(1)
output sacf_type_req from macf(1) via determline to sacf_type_table(1)
> track trans sacf_type_table(1) : start
> track trans sacf_type_table(1) : start
> track trans sacf_type_table(1) : start
> fire sacf_type_table(1) : start
step = 3
trans sacf_type_table(1) : start
> track trans sacf_type_table(1) : from_idle_input_sacf_type_req
> track trans sacf_type_table(1) : from_idle_input_sacf_type_req
> fire sacf_type_table(1) : from_idle_input_sacf_type_req
.
.
step = 38
trans macf(1) : from_waiting_input_m_ind
input m_ind from sacf(1) to macf(1)
output maup from macf(1) via ma_pdu to pdu_maker(1)
> track trans pdu_maker(1) : from_idle_input_maup
> track trans pdu_maker(1) : from_idle_input_maup
> fire pdu_maker(1) : from_idle_input_maup
step = 39
trans pdu_maker(1) : from_idle_input_maup
input maup from macf(1) to pdu_maker(1)
output masend1 from pdu_maker(1) via ma_pdu to macf(1)
> track trans macf(1) : from_pdu_waiting4_input_masend1
> track trans macf(1) : from_pdu_waiting4_input_masend1
> fire macf(1) : from_pdu_waiting4_input_masend1
step = 40
```

```
trans macf(1) : from_pdu_waiting4_input_masend1
input masend1 from pdu_maker(1) to macf(1)
output ap_ind from macf(1) via apline to env
> watch list >> 'inap10_25 watch'
> track >> 'inap10_25.trk'
> quit
end of simulation
```

(그림 11) 사용자 상호작용 시뮬레이션 과정
(Fig. 11) Interactive simulation processes

는 스텝의 일부만 도시하였다. 모든 가능한 경우를 적용하여 시뮬레이션을 한 결과가 (그림 12)에 나타나 있다. 여기서도 사용자 상호작용 시뮬레이션과 마찬가지로 입력 데이터로는 InitialDP와 ConnectToResource를 적용하였다. 여기서 deadlock은 1로, exception은 10으로 제한 하였으며, 결과에서는 deadlock, exception 모두 발생하지 않았다. Transition coverage rate는 52.22%로 43개의 transition이 발생하지 않았고, state coverage rate는 80.36%로 11개의 state에 대해 시뮬레이션이 이루어지지 않았다. 이는 입력 데이터들이 InitialDP와 ConnectToResource에 국한되어 있기 때문인데, 입력 데이터의 범위를 확장함으로써 개선될 부분이라고 본다.

INAP시스템의 일부인 SACF의 유한상태머신을

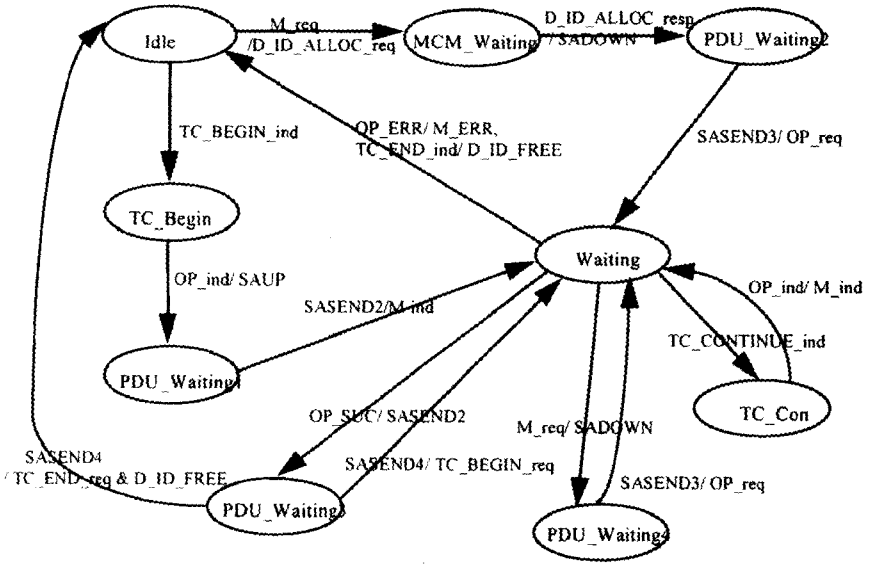
```
> feed apline ap_req(0,10,3,10,10,10,1,1,0,'8612583','8604894')
> feed tc_line tc_continue2_req(19,10,3,0,0,0,0,0,'8612583','8604894',1)
Parameter list length is not equal to signal tc_continue2_req parameter list length
> feed tc_line tc_continue2_req(19,10,3,0,0,0,0,0,'8612583','8604894',1)
> mode breadth
> deadlock limit 1, halt
> exception limit 10
> stop limit 10
> define scc_sink_limit 10
> define states_limit 0
> define depth_limit 0
> define depth_limit_stop   false
> define compose_unit 0
> define compress_unit 0
> verify

(8192 states 40416 transitions 30 seconds, depth=7, breadth=3796)
(16384 states 93671 transitions 69 seconds, depth=8, breadth=5006)
(24576 states 157688 transitions 116 seconds, depth=10, breadth=5446)
.
.
 (2293760 states 7436809 transitions 18682 seconds, depth=49, breadth=598341)
(2301952 states 7464673 transitions 18906 seconds, depth=49, breadth=598341)
(2310144 states 7492445 transitions 19125 seconds, depth=49, breadth=598341)

Number of states : 2318211
Number of transitions : 7519585
Maximum depth reached : 49
Maximum breadth reached : 598341
duration : 322 mn 36 s
Number of exceptions : 0
Number of deadlocks : 0
Number of stop conditions : 0
Transitions coverage rate : 52.22 (43 transitions not covered)
States coverage rate : 80.36 (11 states not covered)
end of simulation
```

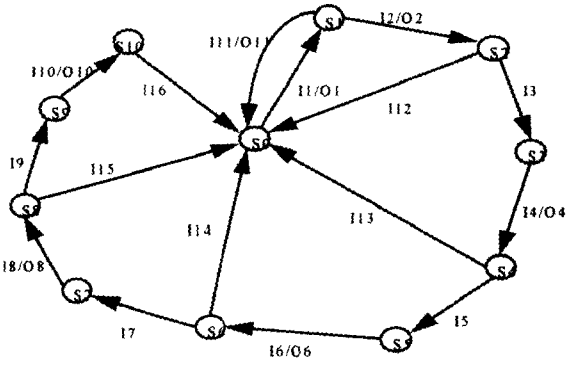
(그림 12) 모든 가능한 경우를 시뮬레이션 한 결과
(Fig. 12) The result of exhaustive simulation

(그림 13)에서 보이고 있으며, 또한 모의 테스트를 통해 검증된 INAP시스템의 AAB서비스 처리 프로세스의 일부인 SACF를 SSF관점에서 SCF와 상호작용하는 절차를 (그림 14)에 상태천이도로 나타내었다. 이 그림들에서 볼 수 있듯이 모든 상태 천이들이 초기 상태로 도달가능하며, 따라서 deadlock이나 livelock이 존재하지 않음을 알 수 있다.



(그림 13) SACF의 유한상태머신

(Fig. 13) SACF Finite state machine



I1 M_req(InitialDP)
I2 OP_SUC
I3 TC_CONTINUE_ind
I4 OP_ind(ConnectToResource)
I5 TC_CONTINUE_ind
I6 OP_ind(DisconnectForwardConnection)
I7 TC_CONTINUE_ind
I8 OP_ind(FurnishingChargingInformation)
I9 TC_CONTINUE_ind
I10 OP_ind(Connect)
I11 OP_ERR
I12 TC_END_ind
I13 TC_END_ind
I14 TC_END_ind
I5 TC_END_ind
I6 TC_END_ind

O1 OP_req(InitialDP)
O2 TC_BEGIN_req(InitialDP)
O4 M_ind(ConnectToResource)
O6 M_ind(DisconnectForwardConnection)
O8 M_ind(FurnishingChargingInformation)
O10 M_ind(Connect)
O11 M_ERR

(그림 14) SCF와 상호작용하는 SSF SACF

(Fig. 14) State transition diagram of SSF SACF interacting with SCF for AAB service

6. 결 론

본 논문에서는 객체지향방법론을 이용하여 IN CS-1의 목표 서비스 중의 하나인 AAB서비스를 대상으로 INAP을 분석, 설계하였으며, 모의 테스트를 통하여 검증하였다. 또한 ITU-T에서 제안하고 있는 차세대 지능망에 적합한 INAP 구조를 도입하여, 이에 추가적으로 기능 요소들을 정의함으로써 INAP설계 구조를 본 논문에서 제안하였다. 객체지향방법론을 도입함으로써, 차세대 지능망이 CS-2/3로 진화함에 따라 프로토콜도 이를 수용할 수 있는 구조로 확장 가능해지고, 유지보수, 재사용성을 용이하게 한다. 이와 같은 방법에서는 서비스와 프로토콜이 서로 독립적이기 때문에 새로운 서비스의 추가나 수정이 어려웠던 초기 지능망의 단점을 해결할 수 있고, 새로운 서비스를 도입할 때 관련된 소프트웨어의 추가 개발 부담이 줄기 때문에 다양한 서비스를 신속하게 도입할 수 있는 장점이 있다. 현 단계에서는 AAB서비스만을 대상으로 하였는데, 향후 대상 서비스 범위를 넓혀 확장함으로써 완전한 형태의 차세대 지능망 응용 프로토콜 구현을 가능하게 할 것이다.

참 고 문 헌

[1] ITU-T Revised Recommendations, Interface Recommendation for IN CS-1, Q.1218, 1995.

[2] ITU-T Baseline Document for IN CS-2, 1995.

[3] ETSI, IN;Global functional plan for IN Capability Set 1, TCR-TR 016, 1994.

[4] ETSI, IN;IN CS 1 Core Intelligent Network Application Protocol Part 1:Protocol specification, ETS 300 374-1, 1994.

[5] M C Bale, "Signalling in the intelligent network", BT Technical Journal, Vol. 13, No. 2, pp. 30-42, April 1995.

[6] 배성용, 조평동, "차세대지능망 프로토콜 기술", 정보과학회지, 제13권 제8호, pp. 54-64, 8월, 1995.

[7] Magdalena Feldhoffer, "Object-Oriented Modeling of the Application Layer Structure", IFIP, pp. 235-247, 1992.

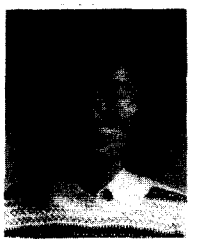
[8] J. Rumbaugh, Object Oriented Modeling and Design, Prentice-Hall, 1991.

[9] ObjectGEODE Method Guideline, Verilog, 1995.

[10] An Introduction to SDL and MSC, Verilog, 1995.

[11] GEODE Editor Reference Manual, Verilog, 1995.

[12] ObjectGEODE Simulator, Verilog, 1995.



도 현 숙

1989년 성신여자대학교 전산학과 졸업(이학사)

1989년~1991년 한국전자통신연구소 근무

1994년 日本 동경공업대학 지능과학전공 졸업(공학석사)

1994년~현재 한국전자통신연구원 지능망구조연구실 근무

관심분야:차세대지능망 응용 프로토콜, 적합성시험, AIN 서비스