인터넷 환경에서의 차세대 지능망 적용을 위한 SCP설계 및 구현

(Design and Implementation of Service Control Point (SCP) for a Next-Generation Intelligent Network based on Internet Environment)

이 지 영 † 김 연 중 †† 마 영 식 † 김 동 호 ††† 안 순 신 ††††
(Jiyoung Lee) (Yeon-Joong Kim) (Youngsik Ma) (Dongho Kim) (Sunshin An)

요 약 지능망은 사용자의 다양한 서비스 요구에 부응하여 여러 가지 서비스를 보다 간단히 개발 적용시키는 것을 목적으로 한다. 이에 따라 차세대 지능망 플랫폼은 빠른 서비스 처리, 다양한 서비스의 동적 적용 등이 가능한 시스템이 되어야 한다. 본 논문에서는 차세대 지능망 플랫폼의 핵심이 되는 SCP(Service Control Point)를 설계하고 구현하였으며, 특히, 설계시 확장성, 동적 서비스 분배 및 인터넷 환경을 고려한 빠른 서비스 처리 등의 지원에 중점을 두었다. 이 SCP는 시스템 운용을 담당하는 전용 커널, 전체 서비스를 관리하는 서비스 관리자, 각 서비스 별로 SLPI(Service Logic processing Program Instance)를 관리하는 SLPI관리자 등으로 구성되어 있으며, 사용자 요청은 각각의 SLPI들에 의해 처리된다. 또한, 구현된 SCP상에서 동작하는 서비스의 손쉬운 생성을 지원하는 SCE(Service Creation Environment)환경을 제시하고, 이를 이용한 서비스 적용 예도 보인다. 현재 구현된 모든 구성 요소들은 유선 인터넷 환경을 고려하여 설계되었으며, 향후 IMT-2000 환경으로의 적용도 가능할 수 있도록 각 부분별로 구분하여 설계하였다.

Abstract Intelligent Network (IN) is an architectural framework for the rapid and uniform provision of advanced services. Therefore, Next-generation IN platform should support fast service handling, the dynamic adaptation of various services according to various requirements of users.

In this paper, we design and implement the Service Control Point (SCP) that is based on the Internet environment. Especially, we consider the service flexibility, the dynamic distribution of service and propose a set of mechanisms to optimize the performance of this platform. This SCP consists of the platform kernel to control internal operations, the service manager to control all of services, SLPI (Service Logic processing Program Instance) manager to manage each service, and SLPI to handle service interaction. In addition, we propose the Service Creation Environment (SCE) to create easily services on the implemented platform and show the example of Call Forwarding service operated on this platform.

1. 서 론

† 학생회원 : 고려대학교 전자공학과
ljy@dsys.korea.ac.kr
mys@dsys.korea.ac.kr
†† 비 회 원 : 고려대학교 전자공학과
kyj@dsys.korea.ac.kr
††† 비 회 원 : 한라대학교 정보통신공학부 교수
imi@hit.halla.ac.kr
†††† 종신회원 : 고려대학교 전자공학과 교수
sunshin@dsys.korea.ac.kr
논문접수 : 2001년 2월 2일
심사완료 : 2001년 5월 16일

다양한 형태의 통신에 의한 정보의 교환이 사회 생활을 영위하는 절대적인 요소로 등장함에 따라 통신 서비스에 대한 사용자 요구 사항이 증가되고 있다. 이를 지원하기 위하여 다양한 사용자 요구 사항에 부응하고 새로운 서비스를 신속히 도입할 수 있는 지능망의 구현 및 보급이 각 국가, 통신사업자 별로 추진되고 있다. 지능망은 새로운 서비스의 도입과 관리를 용이하게 할 수 있으며, 모든 통신망에 적용할 수 있는 구조적 개념이다.

지능망의 목적은 다음과 같이 다섯 가지로 나누어 볼 수 있다. 첫째로, 시장에서 요구하는 새로운 서비스의 개

념을 바로 구현하여 도입할 수 있도록 하는 토대를 제공하고, 둘째로, 현재의 일반적인 음성과 데이타 전송 서비스에서 정보서비스, 광대역 멀티미디어 전송 서비스와 같은 보다 넓은 범위의 다양한 서비스를 제공할 수 있도록 한다. 셋째로, 여러 회사의 장비가 서로 공존하는 환경에서 서로 다른 회사의 장비와도 상호 연동될 수 있는 환경을 제공하고, 넷째로, 이미 광범위에 걸쳐 서비스를 제공하고 있는 현재의 장비가 단시일 내에 교체될 수 없으므로, 현재의 망에서 채택될 수 있는 기술을 토대로 한다. 마지막으로, 서비스 제공자에게 장비 제공자와 독립적으로 서비스를 개발할 수 있는 능력을 부여하고, 또한 망 운용자로 하여금 장비 생산자의 통신망 관련 장비 개발과는 독립적으로 망 내로의 기능과 자원의 적절한 배치 및 망의 효율적 관리를 가능하도록 한다.

지능망의 범위는 통신망의 유형에 따라 공중 전화 교환망(PSTN, Public Switched Telephone Network), 패킷 데이타 교환망(PSDN, Public Switched Data Network), N-ISDN, B-ISDN등 여러 종류의 망에 대한 적용을 기반으로 하고 있고, 서비스의 종류에 따라 부가 서비스를 포함한 여러 종류의 서비스를 지원하며, 기존/미래의 전송 서비스 기능을 이용한다.

지능망은 다음과 같은 특징을 갖는다. 우선, 정보처리 기술과 망 자원을 주로 많이 사용하게 되고, 모듈화된 재사용 가능한 망 기능을 이용함으로써 집약된 서비스의 개발 및 구현이 이루어지며, 모듈화된 기능들을 실제 망 요소로 유연하게 할당할 수 있다. 그리고, 서비스와 독립적인 인터페이스를 통한 망 기능간의 표준화된 통신을 제공하고, 서비스 제공자에게 망 기능의 조합으로 서비스를 통합할 수 있는 기능을 제공한다. 또한, 서비스 가입자에 의한 가입자 및 사용자 관련 서비스 속성의 제어가 가능하게 되고, 서비스 로직의 표준화된 관리가 가능하게 된다. 이러한 특징들을 지원하기 위해서, 지능망 서비스 플랫폼은 서비스의 안정성, 대용량/실시간 처리, 기존 서비스의 변경 및 신규 서비스의 도입이 용이하도록 구현되어야 한다. 여기에서 지능망 서비스 플랫폼이란 요구 서비스에 대한 적절한 서비스 로직을 선택하여 수행하고, 서비스 로직간의 상호작용을 관리하며, 외부 기능 실체에 접근하는 기능을 수행하는 전반적인 지능망 동작 환경을 의미한다.

본 논문에서는 서비스의 동적 적용이 가능하고, 사용자 요청을 빠르게 처리할 수 있도록 최적화된 지능망 플랫폼 의 핵심이 되는 SCP 구조를 설계하고, 그 구현 결과를 제시한다. 모든 구성 요소들은 인터넷 환경을 기반으로 설계 및 구현되었다.

본 논문의 구성은 다음과 같다. 2장에서는 지능망 플랫폼들에 대한 관련 연구를 제시하고, 3장에서는 SCP의 전체적인 구조를 제시한다. 4장과 5장에서는 SCP의 각각에 대한 세부적인 구조와 동작을 기술한다. 6장에서는 구현된 SCP의 검증을 위해 호 전환 서비스를 적용시킨 예를 보인다. 마지막으로, 7장에서 향후 연구 사항을 제시하고 결론을 맺는다.

2. 관련 연구

지능망 서비스를 제공하기 위해 여러 연구에서 지능망 플랫폼이 개발되고 있는데, 대부분의 지능망 플랫폼들은 HW 성능 향상 및 다양한 환경 지원 측면에 중점을 두어 설계되고 있다.

Ericsson Jambala[4] 는 서비스 노드들의 하부 구조를 제공하는 플랫폼으로써, SCP와 HLR 등이 이 플랫폼을 기반으로 동작하게 된다. 이 플랫폼은 TelORB (Telecommunication Object Request Broker)라는 운영체계 겸 미들웨어를 근간으로 한다. SCP(Service Control Point)나 HLR(Home Location Register) 등처럼 실시간과 고장 허용을 요구하는 시스템들은 이 플랫폼 위에서 하나의 응용 프로그램 형태로 구현되며, 서비스 생성 환경(SCE, Service Creation Environment)과 같이 성능에 대한 요구가 적은 응용프로그램들은 외부의 다른 운영체제 위에서 구현된 후, CORBA(Common Object Request Broker Architecture) 2.0 인터페이스를 사용하여 통신할 수 있다. 이 플랫폼은 순수하게 SCP자체의 성능만을 고려한 것이 아니기 때문에, 서비스 제어를 위한 부가적인 오버헤드가 많이 존재할 수 있다. 또한, 외부망과의 통신시에는 CORBA 인터페이스를 사용하기 때문에 기존 시스템들과의 연동을 위해서는 게이트웨이가 반드시 필요하게 되며, 인터넷 환경에 대한 고려도 부족하다.

HP OpenCall Multi-service Controller[5]는 PSTN망과 인터넷망을 통해 향상된 서비스를 제공하기 위한 플랫폼으로써, 서비스 생성 환경(SCE), 서비스 실행 시스템(SEP), 서비스 관리 시스템(SMP)으로 구성되며, 기존 SCP의 기능을 VoIP(Voice over IP)에서도 적용될 수 있도록 하는 서비스 중재 기능도 지원한다. 특히, 서비스 실행 시스템의 경우, 실시간 서비스 처리를 지원하며 기존 서비스들도 실행될 수 있도록 설계되었다.

기존 PSTN망과 인터넷 망 사이의 연동을 위한 새로운 기술들도 도입되고 있는데, 특히 소프트스위치에 대한 연구가 활발히 진행중이다. 소프트스위치는 이기종의 신호방식에 대한 중계 역할 및 여러 형태의 미디어 게

이트웨이를 제어해주는 플랫폼으로써, 개방형 구조, 개방형 표준, 표준화된 모듈화로 인한 확장성과 안정성, 호환성 등을 지향하고 있다. 따라서, 각종 지능망 서비스 및 부가서비스를 쉽게 추가할 수 있고 통합 망관리 기능 등을 사용할 수 있어 수준 높은 시스템 관리가 가능하다. 루슨트 소프트스위치[6]는 기존 회선망 기반 서비스들이 패킷망에서도 그대로 적용될 수 있도록 하는 개방형 플랫폼으로써, 소프트웨어기반 호/서비스 제어 계층을 제공함으로써, 새로운 응용 서비스들 및 다양한 프로토콜들의 변환 등을 손쉽게 적용할 수 있도록 해준다. 3Com 소프트스위치는 호 제어, 미디어 제어 , 시그널링 변환 등을 위한 개방형 시스템 구조를 제공하므로 필요한 기능의 손쉬운 추가가 가능하며, 기능별 분산 구조로 설계돼 장애대처가 용이하다. 시스코 소프트스위치는 시스템 관리를 위한 EMS(Element Management System), 호 처리를 제어하는 CA(Call Agent), 그리고 지능망 서비스를 제공하는 FS(Feature Server), 이 세 가지로 구성되며, 타사 장비와의 연동이 쉽다는 장점이 있다. 이상과 같은 소프트 스위치 관련 기술들은 다양한 망 환경에 대한 중계 및 제어 등을 제공해 주며, 지능망 서비스가 인터넷 망에서 바로 적용될 수 있는 기반 환경이 되어준다.

새로운 사업들에 대한 요구가 증대되면서 서로 상이한 망들이 개방형 API를 지원해야 할 필요성 또한 함께 증대되고 있고, 따라서 이를 위한 다양한 표준들도 제정되고 있다. 통합된 서비스 환경을 위한 JAIN APIs [7]는 기존의 전화망과 인터넷 망에서 이식하기 쉽고, 안정적인 서비스를 가능하게 해준다. 전화망, 패킷망(e.g. IP or ATM)과 기존의 지능망 기반의 서비스 생성을 위한 JAVA 인터페이스를 제공함으로써, 전화망과 인터넷 망을 연동 할 수 있도록 해준다. 더 나아가서, JAVA 응용들이 안전하게 네트웍 자원에 접근 할 수 있게 함으로써, 여러 가지 새로운 서비스들이 보다 신속하게 창출 될 수 있다. Parlay API[8]는 제 3 사업자와 망 운용자들들로 하여금 망 자원에 대한 실시간적 제어가 가능한 새로운 응용들을 만들 수 있도록 해준다. 특히 자바언어의 장점을 Parlay API에 적용하는 JAIN Parlay API들도 정의되고 있는 상황이다. 하지만, 이런 표준들은 개방형 API의 제시 및 전체적 계층 구조 등에 대해서 언급할 뿐, 구체적인 내부 구조 및 동작에 대한 것은 아직까지 제시하지 않고 있다.

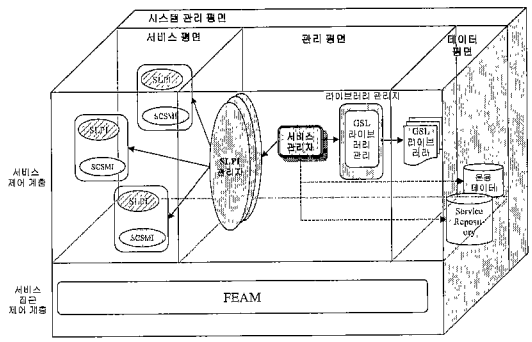
3. 차세대 지능망 플랫폼 구조

본 장에서는 지능망 플랫폼의 전체 구조를 기술한다. 우선 전체적인 개념적 구조를 제시하고, 이를 바탕으로 한 실제적인 구현 구조를 세부적으로 제시하며, 이 구조는 지능망 표준안 CS-2를 근간으로 한다[9,10,11,12]. 본 논문에서는 주로 SCP의 구조에 초점을 맞추어 기술하지만, 이 SCP의 하부 통신 모듈 및 커널 등은 다른 물리 실체의 구현시에도 동일하게 적용되어 구현되었기 때문에, 구현된 시스템을 지능망 플랫폼이란 용어로 통틀어 지칭하여 사용한다.

3.1 개념적인 구조

본 플랫폼은 최적화된 서비스 수행 및 제어를 목표로 설계되었다. 이를 위해 각 서비스 로직들은 각각 자신들의 상태 정보를 함께 가지고 동작하고, 시스템내의 오버헤드를 줄일 수 있도록 송수신 루틴을 간소화하였으며, 전체적인 제어를 위해 꼭 필요한 기능만을 설계에 포함시켰다.

그림 1은 지능망 플랫폼의 전체 구조에 대한 공간적 구성을 나타낸다. 지능망 플랫폼의 전체적 구조를 조망하기 위한 것으로써, 수평적으로는 시스템 제어 평면, 서비스 평면, 관리 평면, 데이타 평면의 네 가지 평면으로 구성되며, 수직적으로는 서비스 제어 계층, 서비스 접근 제어 계층의 두 가지 계층으로 구성되어 있다.



FEAM(Functional Entity Access Manager) : 기능 실체 접근 관리자
SLPI(Service Logic processing Program Instance) : 서비스 로직 처리 프로그램 인스턴스
SCSM I(SCP Call State Model Instance) : SCP 콜 상태 모델 인스턴스

그림 1 지능망 플랫폼의 공간적 구성

주요 구성 요소들의 역할은 다음과 같다.

• 서비스 관리자(Service Manager) : 서비스 제공을 위해 SCP에서 필요한 기능들의 전체적인 제어 및 관리를 담당한다. 각 서비스별로 SLPIM(SLPI Manager)을 생성시키는 역할과 SLPIM에서 보고하는 에러 사항에 대한 처리를 담당한다. 또한, 다른 기능 실체로부터의 입력 메시지를 변환하여, 그것들을 해당 SLPIM에 전달하는 역할을 수행한다. 이러한 동작들은 모두 내부의 서비스 제어자(Service Controller)에 의해 처리된다.

• SLPI 관리자(SLPIM) : 하나의 서비스에 대한 SLPI들을 생성하고, 관리하는 역할을 수행하며, SLPI에서 SDF(Service Data Function)에 대한 접근 동작이 수행되어야 하는 경우, 데이타 베이스 접근 메커니즘을 제공한다. 또한, SLPIM은 자신이 생성될 때, 해당 서비스에 대한 정보를 바탕으로 GSL(Global Service Logic)을 유지하여, SLPI가 참조할 수 있도록 한다. SLPIM에서는 자신이 생성시킨 SLPI에 대한 정보를 지속적으로 유지하며, 필요한 경우, SLPI로 가야 하는 메시지들에 대한 처리도 담당한다. 이러한 동작들은 모두 내부의 서비스 로직 제어자(SLC, Service Logic Controller)에 의해 처리된다.

• SLPI(Service Logic processing Program Instance) : 하나의 서비스 요청 호에 대한 직접적인 처리를 담당한다. SLPIM에서 지정한 GSL에 따라 SIB (Service Independent Building Block)들을 동작시키며, 외부 물리 실체와의 통신 시 필요한 상태 정보 관리는 내부적으로 SCSMI(SCF Call State Model Instance)를 생성시킴으로써 수행한다. 하나의 서비스 호 요청에 대하여 기본적으로 SLPI는 하나가 존재하게 되며, 필요에 따라 다른 SLPI와 상호 동작을 수행할 수 있다.

• 라이브러리 관리자 : SIB 라이브러리와 서비스 정보에 대한 관리를 담당한다. 즉, SIB들에 대한 정보를 가진 SIB 라이브러리, SIB 라이브러리를 구성하기 위해 필요한 FEA 라이브러리 및 IF 라이브러리, 그리고 전체 GSL에 대한 정보를 유지하는 GSL 서비스 정보 등을 관리한다.

3.2 구현상의 세부 구조

본 절에서는 각 구성 요소들간의 상호 관계 및 통신 방법에 대해 설명한다. 그림 2에서는 지능망 플랫폼상에서 각 구성 요소들간의 연관 관계를 보여 주고 있다. 모든 구성 요소들은 하나의 프로세서내에서 동작하며, 각 인스턴스 들은 각각 하나의 쓰레드 들로써 동작하게 된다.

그림 2의 세부구조는 표준안 CS-2에서 제시된 SCF (Service Control Function) 모델의 기본 개념은 따르되, 인터넷 환경에 적합하도록 몇 가지 사항을 차별화하였다.

하부 프로토콜로써, 기존의 SS7(Signaling System 7)프로토콜 대신 TCP/IP를 사용함으로써, 향후 All-IP 망에서 인터넷과 바로 연동될 수 있게 하였다. 즉, 통신 모듈은 TCP/IP의 소켓을 기반으로 하여 연결 관리 및 데이타 송수신 처리를 담당하게 하였고, 내부 메시지 처리 모듈들은 모두 송수신 데이타를 패킷 단위로 처리하도록 구성되어 있다. 이러한 부분들은 기존의 지능망에 비해 연결 호 단위 제어를 좀 더 효율적으로 할 수 있도록 해줌으로써, 인터넷 환경에 더 적합한 구조가 될 수 있다. 또한, TCP/IP 프로토콜 위에는 TCAP, INAP을 구현함으로써, 기존 표준에서 정의된 것들을 그대로 적용할 수 있도록 하였다.

수신된 메시지들은 다시 내부에서 처리하기 쉬운 형태로 변환된 뒤, 적절한 처리 단계들을 거쳐 해당 서비스 로직에게 전달되어, 서비스 제어가 시작되며, 구체적인 사항들 및 특징들은 다음절에서 설명된다.

전체적인 통신 모듈과 제어 모듈들은 분리되어 설계되어 있기 때문에, 통신 모듈만의 변경으로 기존망으로의 이식 또한 가능하다.

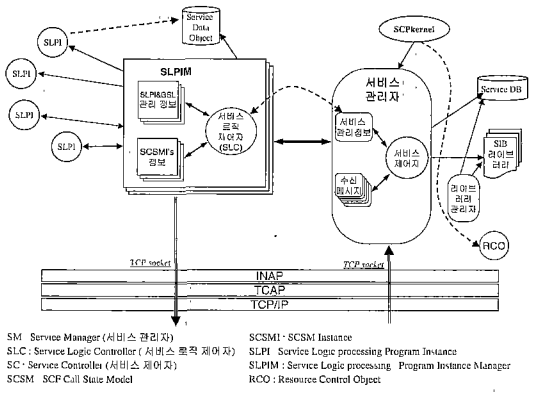


그림 2 지능망 플랫폼의 구현상 세부 구조

그림에서는 하부 통신 모듈은 세 계층으로 구성되어 있음을 보이고 있으며, 그 이외의 각 구성 요소들의 관계는 다음과 같다.

• 각 구성 요소들(서비스 관리자, SLPIM, SLPI 등) 간의 통신은 쓰레드 동기화 함수 들을 사용하여 구현하였다.

• SLPIM내의 SLC(Service Logic Controller)는 각각 쓰레드로 동작하는 SLPI들의 생성, 삭제 등의 관리를 담당하며, 이를 위해 SLPI&GSL관리 정보를 유지한다.

• 서비스 관리자는 지능망 플랫폼내의 전체 서비스를 관장한다. 이를 위해 내부에는 서비스 제어자라는 쓰레드가 동작하여, SLPIM의 생성, 삭제, 서비스 정보 관리, 외부 기능실체 들로부터의 메시지 관리 등을 수행한다.

• 각각의 SLPIM 또는 SLPI들은 데이타 질의어를 통해 서비스 데이타 객체(Service Data Object)에 접근한다.

• SLPI는 상태 전이 정보인 SCSMI를 유지하면서, 상대 기능 실체와의 통신을 지원한다. 이를 위해 다른

기능 실체들로부터의 입력 메시지를 해당 SCSM 이벤트로 변환시켜주는 역할을 수행한다.

• SCPKernel은 시스템 전반에 대한 정보를 유지하며, 각각의 요소에 대한 모니터링 기능을 제공한다. 지능망 플랫폼의 초기화 시 가장 먼저 생성되는 부분으로써, 서비스 관리자의 생성을 책임진다.

• 자원 제어 객체(RCO, Resource Control Object)는 시스템 내에서 사용하는 자원 정보와 관련된 데이타들을 제어하는 역할을 수행한다.

• 서비스 데이타 베이스는 지능망 플랫폼에서 제공하는 각각의 서비스에 대한 정보들을 유지한다. 유지되는 정보는 각 서비스에 대한 GSL 정보이며, 서비스 관리자는 이 정보를 SLPIM 생성시 전달함으로써 SLPIM이 서비스에 대한 GSL을 구성하도록 한다. 서비스 정보 파일은 SCE에서 생성하여 SCP로 전달된다.

• SIB 라이브러리내에는 SLPI에서 사용하는 SIB들에 대한 라이브러리들이 유지된다. 각각의 SIB들에 대한 라이브러리 파일은 모두 공유 라이브러리로 유지되며, 각각의 SLPIM는 필요한 SIB 라이브러리를 동적으로 로딩하여 사용한다.

• 서비스 데이타 객체는 SDF에 해당하는 역할을 수행한다. 개발하는 지능망 플랫폼에서는 SDF를 타 기능 실체로 분리하여 개발하지 않고, 내부적인 데이타 베이스을 이용하여 구성하도록 하였다. 이에 따라 각각의 SLPIM은 서비스 데이타 객체에 접근할 수 있는 기능을 유지한다.

그림 3은 지능망 플랫폼을 구성하는 각 구성 요소들이 어떻게 서로간에 메시지를 주고 받으면서 사용자에게 서비스를 제공하는지 보여주고 있다. 모든 메시지들은 서비스 제어 메시지와 내부 제어 메시지로 나눠진다.

서비스 제어 메시지는 서비스 종류에 따라 다양하며, 그림에서는 초기 서비스 요청 메시지와 서비스 수행 중에 주고 받는 메시지 두 가지로 나누었다. 초기 서비스 요청 메시지의 경우, SLPIM이 SLPI를 생성하기 위해서만 사용하기 때문에 SLPIM 메시지 공유 영역 내에만 저장된다. 서비스 수행 중 주고 받는 메시지의 경우, SLPIM과 SLPI 모두 그 처리 과정에 관여하게 된다.

내부적 제어 메시지의 경우, 서비스 제어자가 SLPIM들을 제어하기 위해 사용하는 메시지, SLPIM와 SLPI간 제어 목적으로 주고 받는 메시지 등으로 나뉜다.

외부 기능 실체로부터 수신되는 메시지는 모두 서비스 관리자내의 서비스 제어자에 의해 모두 수신되어 적절한 처리를 거친 후, 해당 SLPIM영역으로 전달되거나, 필요에 따라서는 SLPI메시지 공유영역으로 직접 전달된다.

4. 구성 요소별 세부 구현

지능망 플랫폼을 구성하는 요소로는 서비스 구성 요소들을 관리하는 서비스 관리자, SLPI들을 제어하는 SLPIM, 직접적으로 서비스 요청을 처리하는 SLPI가 있다. 그리고 흐름 제어 같은 비동기 동작은 서비스 관

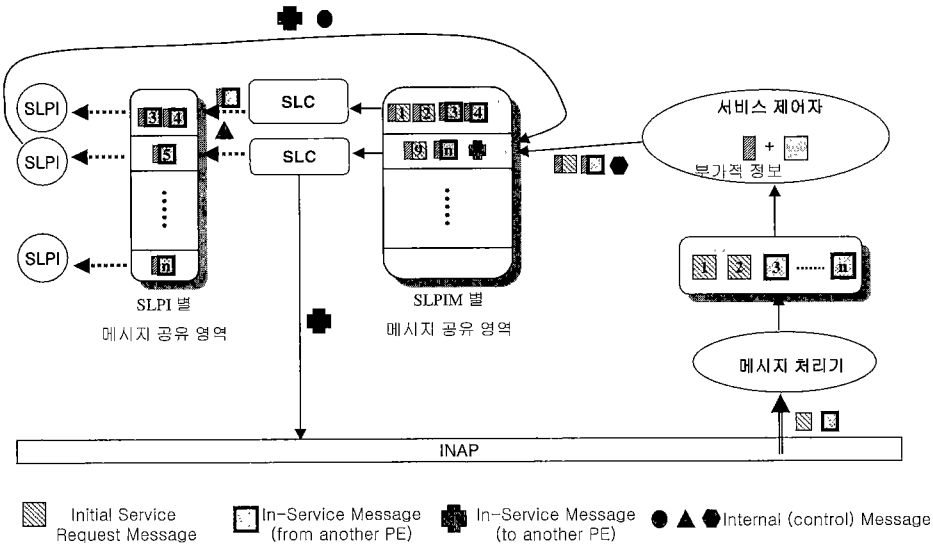


그림 3 각 구성요소들간의 메시지 전달 구조

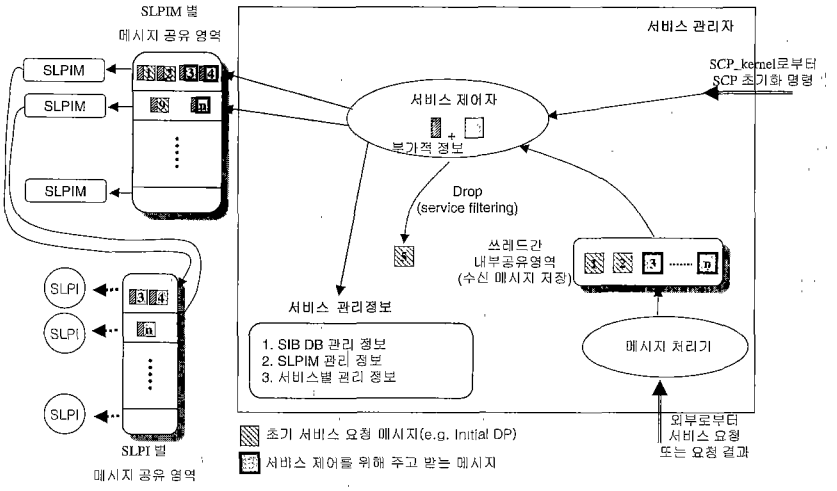


그림 4 서비스 관리자의 구조

리자가 지원하고, 각 SLPI들에 대한 상태 전이는 SLPI 자신이 유지한다. 본 장에서는 이러한 각 구성 요소들에 대한 구조 및 구현 코드의 일부를 제시한다.

4.1 서비스 관리자

4.1.1 서비스 관리자의 기능

서비스 관리자는 지능망 서비스 플랫폼내의 전체 구성 요소들에 대한 제어를 수행하며, 시스템 초기화시 SCP 커널에 의해 시작된다. 이 서비스 관리자는 서비스 제공을 위한 모든 정보를 유지하며, SLPIM에 대한 직접적인 제어 기능을 수행한다. 서비스 관리자의 기능을 나눠보면 크게 세 가지로 나눌 수 있다.

첫째로, 플랫폼 초기화를 담당한다. SCP 커널에 의해 자신이 초기화된 후, 라이브러리 관리자를 통해 읽어 들인 GSL정보들(서비스 정보, SIB 라이브러리)을 기반으로 SLPIM들을 생성시킨다.

둘째로, 플랫폼 최적화를 위해 프로세스, 쓰레드의 동적 생성, 삭제, 관리를 담당한다. 즉, 현재 시스템 상황을 판단하여, SLPIM들을 미리 생성시켜둠으로써 빠른 서비스 지원이 가능하도록 하고, 불필요하게 많이 생성되어있을 경우에는 적절한 수준으로 유휴중인 것들을 삭제한다.

이외에도 서비스 관리자는 서비스 데이타 객체를 통해 서비스 데이타 접근 관리를 수행한다. 즉, 이 객체를 통해, 서비스 관리자는 SCF 동작을 위해 필요한 내부적 공유 데이타, 또는 서비스의 수행 중 외부 기능 실체에 접근시 필요한 해당 서비스 제공 기능 실체들에 대한 정보 등을 제어한다.

4.1.2 서비스 관리자의 구조

4.1.2.1 서비스 관리자의 내부 구조

그림 4는 서비스 관리자의 내부 구조를 보이고 있다. 서비스 관리자는 전체 GSL과 관련된 정보를 유지하면서 SLPIM 생성시 이를 이용하고, SLPIM과 관련된 정보들은 내부 관리 영역과 서비스 정보 공유 영역에 유지한다. 특히, SLPIM, SLPI들과 관련된 연계 번호들을 서비스 정보 공유 영역에 보관함으로써, 서로간의 원활한 정보전달이 가능하도록 하였다.

서비스 관리자는 내부적으로 두 가지 종류의 정보를 유지한다. 첫째는 모든 서비스들과 관련된 GSL 정보를 유지하고, 둘째는 서비스 로직 관리 정보(SLPIM 관리 정보), 라이브러리 관리자에 대한 관리 정보 등과 같은 구성 요소 관리 정보를 유지한다.

그림 5는 서비스 관리자, SLPIM, 그리고 SLPI들간의 메시지 전달을 위해 필요한 공유영역의 구조를 보이고

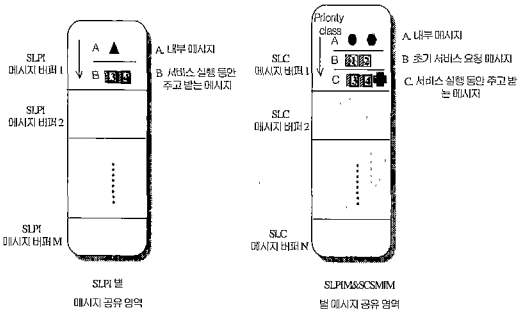


그림 5 메시지 공유 영역의 세부 구조

있다.

SLPIM공유영역은 SLPIM별로 나누어져 있고, 하나의 SLPIM 메시지 영역은 다시 메시지의 종류에 따라 우선순위별로 세 가지, 즉, 내부 메시지, 초기 서비스 요청 메시지, 진행중인 서비스 관련 메시지로 나뉜다. 이렇게 함으로써, 내부 제어메시지를 가장 먼저 처리할 수 있도록 하였다.

SLPI 공유영역도 SLPI별로 나누어져있으며, 하나의 SLPI 메시지 영역은 메시지 종류에 따라 두 가지, 즉 내부 메시지, 진행 중인 서비스 관련 메시지로 나뉜다.

4.1.2.2 서비스 관리 정보 영역의 정의

그림 6은 서비스 관리자 내에 유지되는 서비스 관리 정보 영역이다. 대부분의 갱신은 서비스 관리자에 의해 수행되며, 일부 정보는 SLPIM에 의해서 갱신하게 된다. 즉, 서비스 관리 정보 중 SLPIM 관리 정보의 내용은 SLPIM과 서로 공유하게 되며, SLPI와 관련된 정보들은 SLPIM내에 있는 SLC가 관리하게 된다.

SIB 데이타 베이스 관리 정보 내에는 SIB와 관련된 라이브러리 정보, 버전 정보 등이 포함된다.

SLPIM 관리 정보 내에는 SLC, SLPI 등과 관련된 다양한 내용이 보관되며, 이 정보는 서비스 관리자내의 서비스 제어자와 SLPIM내의 SLC간에 서로 공유된다. SLC와 관련된 정보들은 서비스 제어자가 플랫폼 초기화 시 기록하며, SLPI와 관련된 정보들은 SLC가 새로운 서비스 요청이 들어올 때마다 기록한다. 서비스 관리자는 일반적으로 SLPI와 관련된 정보의 일부만 알면 되지만, 빠른 전송 처리를 위한 확장 시 빠른 메시지 전달을 위해서 나머지 정보들을 사용할 수도 있다. 그림에서 보이듯이 하나의 SLC는 다수 개의 SLPI들을 관리하게 됨으로, SLC 번호 하나마다 여러 개의 SLPI 번호가 결부됨을 볼 수 있다.

서비스별 관리 정보에는 각 서비스별 생성 쓰레드들의 수 및 GSL 로직 포인터를 유지한다.

1. SIB DB 관리 정보

SIB 정보	라이브러리 정보			
	library pointer	SIB Version	SIB State	Reference counter

2. SLPIM관리 정보(SLPIM과 공유되는 정보)

SLC Number	SLC threadID	SLC Message buffer pointer	SLC state	SLPI Number	Mode	SLPI threadID	SLPI Transaction ID	SLPI Message buffer pointer	SCSMI 정보
1	122		new	1	first	10			
				2	fast	18			
				3	fast	20			
2	25		new	1	normal	35			
				2	fast	36			
				9	fast	37			
3	100		new						

그림 6 서비스 관리 영역의 구조

4.2 SLPIM(SLPI Manager)

4.2.1 SLPIM의 기능

SLPIM은 기본적으로 지능망 서비스 플랫폼이 제공하는 각 서비스별로 하나씩 생성된다. 이 SLPIM은 해당 서비스에 대한 요청시 각 요청마다 하나씩의 SLPI를 생성시킴으로써, 서비스 제어를 하게 된다.

SLPIM에 의해 관리되는 SLPI들에게 전달되는 모든 데이타들은 기본적으로 해당 SLPIM이 모두 관리하게 된다. 즉, SLPIM은 각 SLPI들에게 오는 데이타를 자신이 받은 뒤, 적절한 SLPI에게 전달해 주게 되는 데, 필요에 따라서는 초기 제어 메시지만 SLPIM이 받고 나머지 메시지들에 대해서는 SLPI들이 직접 받게 할 수도 있다.

SLPIM은 자신이 제어하는 SLPI들에 대한 정보를 유지하고, 이런 정보를 서비스 관리자와의 통신을 통해 유지함으로써 효율적인 서비스 관리가 가능하도록 한다.

4.2.2 SLPIM의 구조

그림 7에서, SLPIM의 구조는 크게 SLC, GSL관리 모듈로 나뉜다. SLC는 SLPIM 생성시 내부 초기화를 담당하여, GSL 관리 모듈과 통신 모듈의 초기화를 수행한다. GSL 관리 모듈은 GSL에 대한 정보 및 GSL 내의 각 SIB 들에 대한 정보를 SIB 라이브러리로부터 동적 로딩하여 관리하는 부분으로써, 각 오퍼레이션들에 대한 함수 포인터를 유지하게 되며, 이 정보들은 SLPI가 생성될 때마다 그 정보에 대한 포인터를 넘겨줌으로써, 여러 SLPI들이 공유할 수 있도록 한다.

SLPIM에서 내부적으로 유지하게 되는 SLPI관리 정보로는 SLPI 상태 정보 테이블, SLPI 인스턴스 번호, 각 SLPI와 매칭되는 트랜잭션 ID 정보 등이 있다. 서비스 관리자는 GSL Info를 기반으로 SIB 라이브러리로부터 모든 정보를 동적 로딩해서 보관하게 되고, SLPIM의 생성시에 해당 GSL Info를 넘겨주게 된다. 따라서, SLPIM은 사용자 요청 수신시, SLPI생성과 더불어 이 정보도 함께 넘겨줌으로써, 서비스 로직이 수행될 수 있도록 한다.

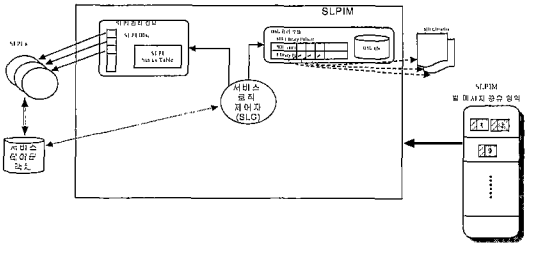


그림 7 SLPIM의 구조

4.3 SLPI

4.3.1 SLPI의 기능

SLPI는 실제로 서비스 로직을 수행하는 주체가 되는 부분이다. 서비스 요청이 들어오면 각 요청마다 하나씩의 SLPI가 생성되며, 생성된 SLPI는 자신의 GSL에 따라 SIB 오퍼레이션들을 수행하여, 상대 기능 실체들에게 적절한 서비스를 제공하게 된다.

서비스 로직 수행자는 GSL 인스턴스를 통해 전체 로직을 수행시킨다. 서비스 로직 수행에 관련된 세부적인 부분은 다음 장에서 자세히 언급된다. SLPI내에 포함되어 있는 통신 모듈은 SLPIM과의 공유 영역 부분으로부터 데이타를 읽어 들이거나, 외부 물리 실체에게 데이타를 전달하기 위한 용도로 쓰인다.

4.3.2 SLPI의 구조

그림 8에서는 SLPI가 SLPIM과 비슷하게 통신 모듈과 GSL 모듈로 구성되어 있음을 보이고 있다. SLPI 동작의 대부분은 서비스 로직 수행자에 의한 서비스 제어에 있으며, 통신 모듈 부분은 각 SIB op내의 IF 수행시에만 사용된다.

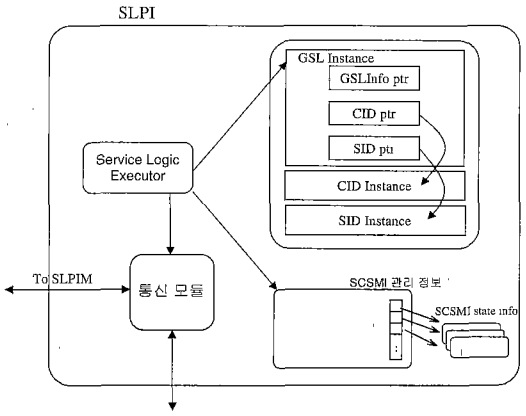


그림 8 SLPI 구조

4.4 라이브러리 관리자

4.4.1 라이브러리 관리자의 기능

라이브러리 관리자는 전체GSL에 대한 정보를 유지하는 GSL Service Info, GSL을 구성하는 SIB들에 대한 정보를 가진 SIB 라이브러리, SIB 라이브러리를 구성하기 위해 필요한 FEA 라이브러리와 IF 라이브러리 등을 관리한다.

SMS(Service Management System)로부터 새로운 서비스의 도입이나 변경 지시가 올 경우, 서비스 관리자는 이를 라이브러리 관리자에게 전달한다. 라이브러리 관리자는 서비스 관리자의 제어에 따라 기능 실체 접근 관리자를 통해 이 정보를 수신한 후, 라이브러리들의 동적 추가, 동적 삭제, 갱신 등을 수행한다. 라이브러리 관리자는 SCE나 시스템 운용자와 직접 상호 작용 기능을 수행할 수도 있다.

라이브러리 관리자는 수신된 라이브러리들을 추가하기 전에, 수신된 로직들이 정상적으로 동작될 수 있는 것인지의 여부와 라이브러리들의 무결성 등을 내부적으로 검사해 본다. 만약, 서비스 로직(GSL)만의 추가라면, 이 로직을 구성하는 SIB들의 연관 관계 및 전체적 흐름만을 해석해서 검증하면 된다. SIB자체에 대한 새로운 추가시에는 단순한 연관관계만을 보는 것이 아니라, SIB 내부적 부분까지 해석한다. 즉, SIB의 입출력의 정확성 여부, SIB를 구성하는 FEA들과 IF들의 적합성 여부, 새로 설계된 SIB와 기존 SIB사이의 동질성 여부, SIB들 간의 충돌 여부, 그리고 기존 FEA, IF라이브러리들과의 충돌 여부 등 다양한 조건들을 검사해야 한다.

이 검사가 통과되면, 새로운 라이브러리의 추가 부분을 확인하여 그에 대한 정보를 해당 라이브러리에 추가한다. 만약, 추가될 로직들의 일관성에 문제가 있다면, 이를 서비스 관리자와 상대 SMF에게 알려준 뒤, 수신된 데이타를 삭제한다.

4.4.2 라이브러리 관리자의 구조

그림 9는 라이브러리 관리자 및 라이브러리들의 구성을 보이고 있다.

각 GSL에 대한 DSL은 POI(Point Of Initiation)와 POR(Point Of Return) 사이에서 이어지는 SIB, 그리고 SIB의 결과에 따라 순차적으로 실행되는 다음 SIB로 구성된다. GSL Service Info 내에는 여러 개의 GSL

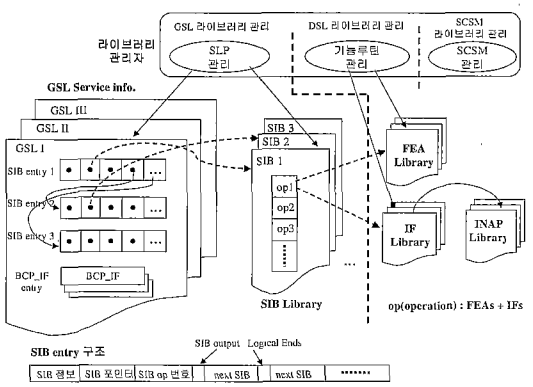


그림 9 라이브러리 관리자와 라이브러리들의 구조

구성 정보들이 들어있다.

각 GSL 구성 정보 내에는 각각 내부에 해당 서비스 로직을 구성하는 데 필요한 SIB관련 정보를 SIB entry 형태로 갖게 된다. 각 SIB entry내에는 현재 SIB에 대한 정보, SIB 라이브러리 내의 포인터, SIB내의 오퍼레이션 번호, 다음 SIB 인덱스 등을 갖고 있다. 다음 SIB가 어떤 것이 될지는 현재 수행 중인 SIB 동작의 결과로써 결정되며, 이 판단 조건은 그림에서처럼 각 next SIB 바로 앞의 SIB output부분에 기술된다. 각 SIB마다 수행 결과인 Logical Ends의 개수가 다르므로, next SIB의 개수도 이에 따라 결정된다.

SIB 라이브러리 내의 각 SIB들은 여러 개의 오퍼레이션들로 구성되어 있다. 각 오퍼레이션들은 각각 여러 개의 FEA들과 IF들로 구성되어 있다. 앞서 언급된 SLP entry내의 SIB op 번호는 바로 해당 SIB내의 특정 오퍼레이션을 가리키는 번호가 된다. 이 구조를 이용하여 GSL을 구성할 경우, 새로운 SIB의 추가나 GSL 변경시, 각 엔트리의 함수 포인터를 바꿔줌으로써 새로운 SIB를 바로 적용할 수 있다.

5. 지능망 플랫폼의 전체적인 수행 과정

본 장에서는 지능망 플랫폼의 전체적인 수행과정 중 초기화 과정 및 서비스 생성 과정을 다룬다.

5.1 플랫폼 초기화 과정

그림 10은 지능망 플랫폼의 초기화 과정을 보여주는 것으로써 세부적인 설명은 다음과 같다.

① SCP 초기화 명령을 받는다.

② 서비스 관리자는 라이브러리 관리자를 통해, 현재 지원되는 서비스에 대한 정보를 읽고, 그에 대한 라이브러리의 정보를 확인한 후, 동적 로딩한다.

③ 서비스 관리자는 서비스의 개수만큼 SLPIM 별 공유 메시지 영역을 생성시킨다.

④ 서비스 관리자는 과정2에서 얻은 서비스 정보를 바탕으로, 각 서비스에 대해 하나씩의 SLPIM들을 생성한다.

⑤ 각각의 서비스에 대한 SLPIM의 생성을 마치면, 생성된 SLPIM의 정보를 서비스 관리 정보 영역에 기록한다.

이상과 같은 과정을 거치면 지능망 플랫폼의 초기화 과정이 종료되며, 이때부터 서비스 관리자는 SSP로 부터의 서비스 요청을 받아들일 수 있게 된다.

5.2 메시지(SSP에서의 호 요청) 처리 과정

그림 11은 SSP로부터 초기 서비스 요청이 들어온 경우에 대한 처리 과정으로써 세부적인 설명은 다음과 같다.

① SSP에서 지능망 서비스에 대한 요청이 전달된다.

② 서비스 관리자는 서비스 관리 정보를 바탕으로 수신한 서비스 요청을 분석한다.

③ 현재 해당 서비스를 플랫폼 내에서 처리할 수 있는지의 여부를 판단한다. 과부하 상태인 경우에는 서비스 요청을 누락시키고, 그렇지 않으면 과정4로 간다.

④ 서비스 관리자는 서비스 요청 메시지를 SLPIM공유 영역에 넣는다.

⑤ SLPIM 내의 SLC는 해당 메시지를 가져온다.

⑥ 가져온 수신 메시지를 이용하여 해당 서비스 요청에 대한 SCSMI 상태 정보를 생성하고, 해당 서비스 호 요청에 대한 트랜잭션 ID를 생성하여 추가적인 SSP와의 상호동작에 사용하도록 한다. 그리고, 새로 생성될 SLPI를 위한 메시지 공유 영역을 생성한다.

⑦ SLPI를 생성한다.

⑧ 생성된 트랜잭션 ID 정보를 포함한 여러 정보들을 서비스 관리 정보 영역에 저장하여 서비스 관리자가 알 수 있도록 해준다.

⑨ SLC는 외부 기능실체에게 요청 수락과 관련된 메

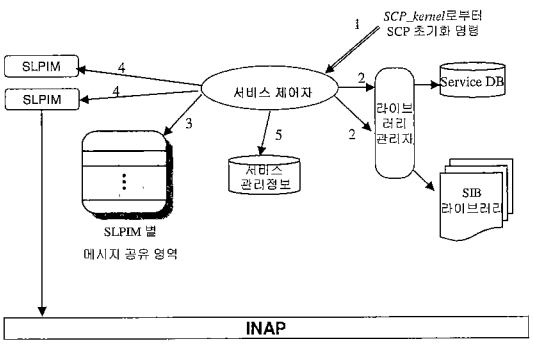


그림 10 플랫폼 초기화 과정

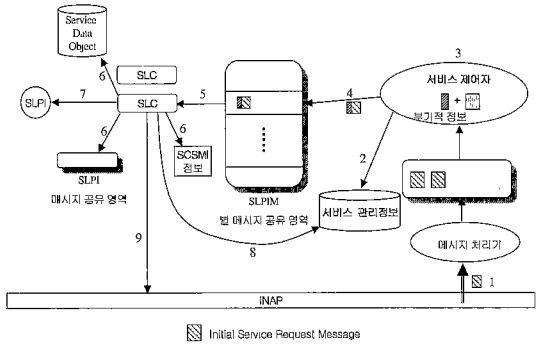


그림 11 메시지(SSP에서의 호 요청) 처리 과정 1

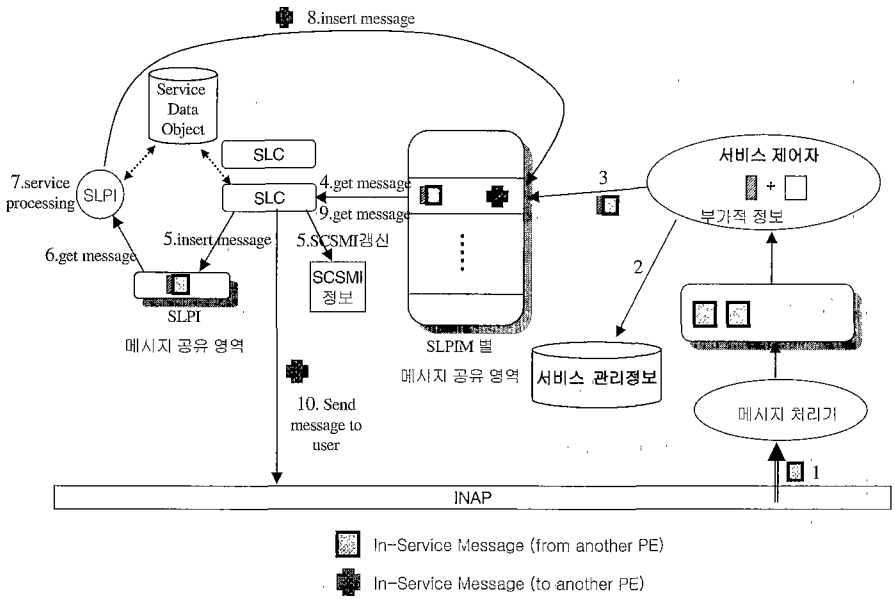


그림 12 메시지(SSP에서의 호 요청) 처리 과정 2

시지를 보낸다.

그림 12는 그림 11의 과정을 거쳐 연결된 상태에서 서비스가 처리되는 과정을 보이고 있다. 일단 시작된 서비스에 대해서는 중단되면 안되므로, 수신되는 즉시 해당 SLPIM 공유영역으로 전달해준다. SLPIM은 이 메시지를 다시 SLPI 메시지 공유영역으로 보냄으로써 SLPI가 사용할 수 있도록 해준다. 그림 12의 과정 ①부터 과정 ⑥까지는 이것을 보여준다. 또한, 과정 ⑦부터 과정 ⑩까지는 SLPI가 요청을 처리한 후, 상대 기능 실체에게 메시지는 보내는 단계를 보이고 있다. 모든 메시지의 변환은 SLPIM이 수행하므로, SLPI는 모든 메시지를 SLPIM 공유영역내에 넣는다.

이상과 같은 과정을 통해 하나의 서비스 요청에 대한 처리 과정이 수행되며, 위의 과정에서 외부 기능 실체에서 SLPI로 데이타가 전달될 경우에는 서비스 관리자 SLPIM SLPI의 순서로 데이타가 전달되며, SLPI에서 외부 기능 실체로 데이타가 전달될 경우에는 SLPI SLPIM 외부 기능 실체의 순서로 데이타가 전달된다.

5.3 새로운 서비스 추가 과정

지능망 플랫폼은 기존 서비스를 제공하는 동안에 새로운 서비스를 제공할 수 있는 구조로 되어 있으며, 그림 13은 새로운 서비스를 도입하는 과정을 보여준다. 새로운 서비스를 도입하는 과정에 대한 자세한 설명은 다음과 같다.

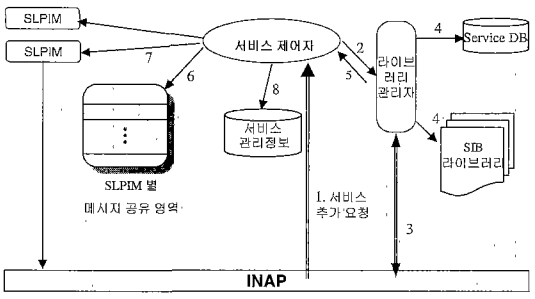


그림 13 새로운 서비스 추가

① SCE 또는 SMS를 통해 새로운 서비스에 대한 정보가 서비스 관리자에게 전달된다.

② 서비스 관리자는 라이브러리 관리자에게 새로운 서비스를 추가하도록 지시한다. 새로운 서비스에 대한 정보는 추가의 SIB가 정의되지 않은 경우, GSL에 대한 정보만이며, SIB가 추가된 경우에는 추가의 SIB에 대한 정보도 포함된다.

③ 라이브러리 관리자는 SMS로부터 새로운 라이브러리 데이타들을 수신한다.

④ 새로운 정보들을 서비스 데이타베이스 와 SIB 라이브러리 내에 추가한다.

⑤ 라이브러리 관리자는 새로운 서비스에 대한 추가가 성공적으로 완료됐음을 알리고, 생성된 추가 정보를

전달한다. 이때 서비스 관리자는 해당 서비스의 라이브러리들을 동적 로딩한다.

⑥ 서비스 관리자는 SLPIM 메시지 공유 영역을 생성한다.

⑦ 동적 로딩된 정보를 이용하여, 새로운 서비스를 위한 SLPIM을 생성한다.

⑧ 새로운 SLPIM을 생성한 후, 생성된 SLPIM의 정보를 서비스 정보 공유 영역에 기록한다.

6. 서비스 적용 예

지능망 플랫폼을 구현하여 그 동작을 검증하기 위해서는 SCP의 구현 외의 다른 물리 실체들(SSP, IP, SDP)도 개발되어야 한다. 이러한 물리 실체의 개발은 구현된 SCP를 검증하기 위해 필요한 동작을 수행하는 물리 실체들의 최소 기능을 구현할 필요가 있다.

본 논문에서는 호 전환(CF, Call Forwarding) 서비스를 이용하여 지능망 서비스 플랫폼을 검증하였다. 즉, 호 전환 서비스를 텍스트 모드로 지속적으로 반복 수행시켜, 지능망 서비스 플랫폼이 지속적인 부하(대량호 등)에 정상적으로 동작하는 지를 검증하였다.

본 장에서는 전체적인 서비스 구현 환경을 제시하고, 호 전환 서비스의 실제적인 적용 예 및 결과를 보인다. 주변의 각 물리 실체들에 대한 실제적인 구현은 본 논문의 범위에 포함되지 않으므로, 더 이상 언급하지 않는다.

6.1 서비스 운용 환경

지능망 플랫폼의 구현 환경은 그림 14와 같다. 사용자 전화 단말을 포함한 모든 물리 실체들은 Solaris 5.7을 운영체제로 사용하는 SUN Ultra 10 환경 (UltraSparc 433 MHz 프로세서, 128Mbytes RAM, E-IDE 방식 하드디스크) 하에서 구현되었으며, 기본적인 통신 인터페이스는 모두 TCP/IP를 사용한다. 각 물리 실체들은 모두 각각 하나의 워크스테이션상에서 동작함을 기본으로 한다. 이 운용 환경에서 사용된 SSP는 본 서비스 적용에의 검증을 위해 최소 기능만을 구현한 시스템으로써 [2], TCP/IP기반으로 하며, 기본적인 호 처리 기능만을 지원한다. 전화기는 SSP와 내부적으로 정의된 프로토콜에 따라 제어를 받게 되며, SSP를 통해 음성, 데이타 등의 송수신이 가능하다. 지능형 정보 제공 시스템(IP, Intelligent Peripheral)은 안내 방송과 같은 자원을 저장하며, INAP 메시지를 통해 SCP의 제어를 받게 된다. 사용자는 이 IP를 통해 안내 방송을 음성으로 또는 문자 정보로 받을 수 있으며, 서로 연결된 사용자 간에는 음성, 또는 문자를 주고받을 수 있게 된다. SCP를 제외한 주변의 물리 실체들은 모두 표준의 일부만을 따르거나 기능 검증만을 위해 자체 구현되었다.

6.2 서비스 생성 환경

본 절에서는 서비스 생성시 사용된 SCE(Service Creation Environment)의 구현에 대해 간략히 설명하며, 구체적인 사항들은 본 논문의 범위를 벗어나므로 생략한다. SCE의 전체구조는 크게 SIB info class, SIB parameter structure 그리고 GUI environment으로 나눌 수 있으며, 각 모듈들의 역할은 다음과 같다.

• SIB INFO Classes : 해당 SIB에 대한 위치정보, SIB들간의 연결정보, SIB의 상태정보를 가지고 있다. 서비스 설계자는 GUI를 통해 원하는 정보를 수정할 수 있다.

• SIB Parameter Info Structures 해당 SIB의 여러 가지 파라미터 입력 값을 저장하고 있다. 서비스 설계자는 파라미터 입력 창을 통해서 파라미터 값을 입력하고, 이 정보는 서비스 정보코드를 생성하는데 참조된다.

• GUI 환경 : 서비스 개발자가 서비스 코드를 작성할

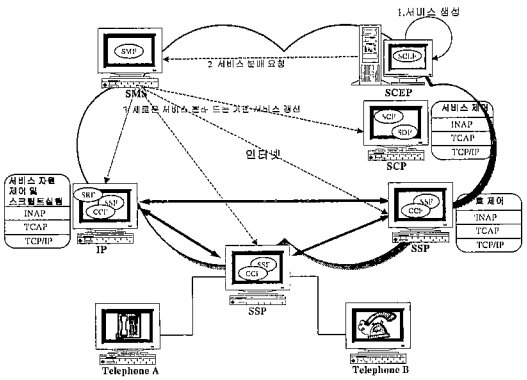


그림 14 서비스 동작 환경

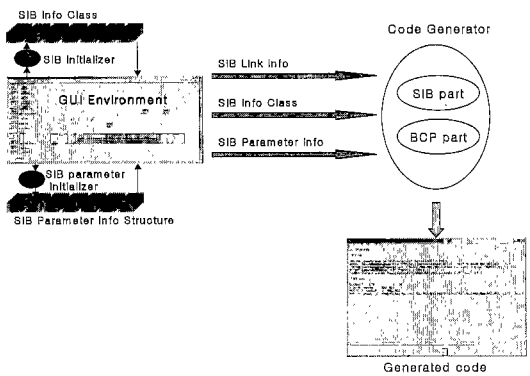


그림 15 SCE의 전체구조

수 있게 해 준다. 설계자는 이 도구를 이용하여, SIB들을 배치한 후 각각의 SIB를 연결하고, SIB의 파라미터를 입력해서, 원하는 서비스 정보 코드를 생성할 수 있다.

6.3 호전환 서비스 적용

6.3.1 서비스 구성 (GSL)

호 전환 서비스는 서비스 가입자에게 걸려 오는 전화(착신호)를 전화를 건 사람(발신자)이 요구한 착신 번호와는 달리 착신자가 등록한 다른 착신 번호로 호를 전환시키는 서비스이다. 본 논문에서 적용한 서비스는 호 전환 서비스를 응용하여 발신자의 호를 전환된 착신 번호에 연결하기 전에 발신자에게 호 전환 여부를 질의하도록 하였다. 이 방법은 만일 전환된 착신 번호가 국외 번호와 같이 기본 전화 요금 외에 추가적인 과금을 수행할 경우에 사용될 수 있다.

호 전환 서비스에 대한 GSL구성은 그림 16과 같다. 모든 SIB를 화면상에 배치하고, 파라미터 입력, 연결정보 입력을 끝낸 후에, 최종적으로, 서비스의 이름과 서비스 Key를 입력해서 서비스 정보 파일을 생성한다.

입력 후에는 생성된 서비스 정보 파일을 새로운 화면으로 출력한다. 출력 파일은 화면과 파일로 동시에 생성되어지며, 서비스키 입력 창에서 입력한 서비스 이름을 갖게 되고, 확장자는 .dat를 갖는다.

지능망 서비스 플랫폼은 서비스별로 생성된 이런 파일들을 기반으로 하여 해당 서비스관련 라이브러리들을 동적 로딩하여 서비스를 생성하게 된다.

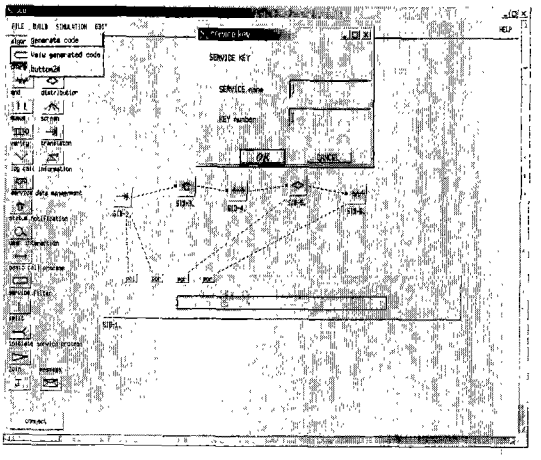


그림 16 SCE환경에서 호전환 서비스 구성

6.3.2 호 전환 서비스에 대한 정보 흐름

호 전환 서비스에 대한 간단한 정보 흐름은 그림 17과 같다.

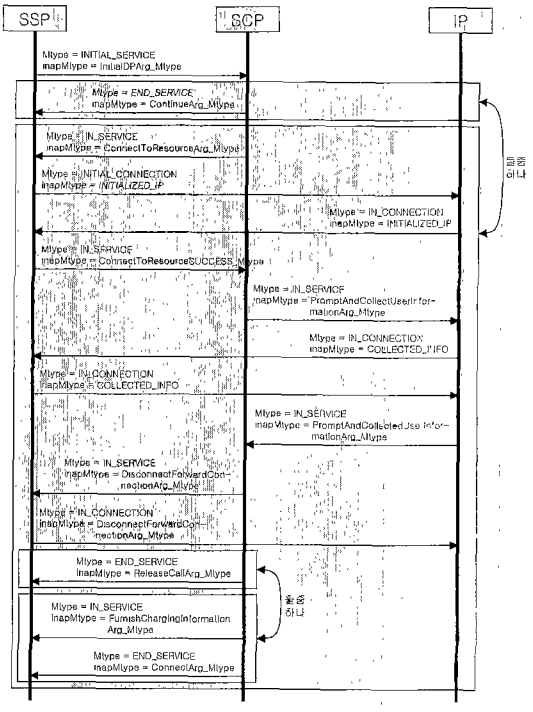


그림 17 호 전환 서비스에 대한 정보 흐름

6.3.3 서비스 동작 방법 및 검증 결과

서비스는 두 가지 모드, 즉 GUI를 이용한 단일호 모드, 그리고 파일을 이용한 대량호 모드로 동작시켰으며, 간이 SSP(-검증을 위해 필요한 기본적 기능만을 갖도록 구현된 SSP) [2], 간이 IP(Intelligent Peripheral)를 활용하여 주변 환경을 구축하였다.

단일호 모드에서는, 우선 GUI를 이용해서 사용자 단말기를 만들고 그 단말을 이용해서 사용자로부터 호에 관한 정보를 받아들이게 했다. 이 모드의 경우는 음성 지원이 가능해서 벨소리나 안내방송이 가능하며, 발신측과 착신측의 통화도 가능하다. 대량호 모드의 경우, 지능망 플랫폼의 처리 성능을 검증해보기 위해 스크립트 파일내에 호와 관련된 필요 정보들을 미리 입력한다. 이 스크립트 파일은 다수의 사용자 요청이 발생되는 것처럼 동작하게 되고, 지능망 플랫폼은 사용자의 직접적 입력대신 이 정보를 분석하여 호처리를 하게 된다. 대량호 처리를 위해서 각 호마다 쓰레드를 탄생시킴으로써 가상적으로 동시 다발적 사용자 요청을 만든다. 간이 SSP와의 통신은 개별적으로 쓰레드간의 통신으로 이루

어진다.

두 경우, 모두 본 지능망 플랫폼상에서 정상적으로 동작하였다. 특히, 대량호 모드의 경우 현재 초당 평균35개의 동시 요청을 처리할 수 있으며, 주변 물리 실체들을 가상적으로 구성 시키는 경우에는 초당 평균 100개 이상의 동시 요청을 처리할 수 있다. 이러한 수치들은 실제 환경에 적용하기에는 상대적으로 낮은 값들인데, 그 이유는 플랫폼 자체의 성능 문제이외에도 SSP나 IP 같은 간이 물리 실체들 자체의 오버헤드, 운용 시스템 자체의 속도 등이 큰 성능 저하 요인으로 작용하였기 때문이다. 따라서, 검증에 사용된 주변 물리 실체들을 실제적인 장치들로 대체하고, 동작 환경의 개선 등을 거친다면 많은 성능 향상을 보일 것이다.

7. 결론 및 향후 연구 과제

본 논문에서는 인터넷(All-IP) 환경 하에서 지능형 서비스 제공을 위한 지능망 플랫폼, 특히 SCP에 중점을 두어 세부 설계하고, 구현하였다. 지능망 플랫폼들의 각 구성 요소들을 모듈화하였으며, 서비스를 관리하는 부분과 통신을 관리하는 부분을 분리하여 설계함으로써 다양한 통신 환경의 적용이 가능하도록 하였다. 사용자 요청의 빠른 처리를 위하여 내부 처리 루틴들을 간략화 함으로써, 시스템 부하를 줄이고 전체적인 처리 속도를 높였다. 또한, 인터넷 환경에 적합하도록 하부 통신 모듈들을 TCP/IP 소켓 기반의 관리 구조로 설계하였고, 내부의 메시지 처리 루틴들을 패킷 단위 처리가 가능하도록 구현하였다.

구현된 지능망 플랫폼의 검증을 위해 간략한 서비스들도 구현하여 적용시켜 보았다. 현재 적용된 서비스들로는 호전환 서비스 및 범용 개인 통신 서비스이 있으며, 플랫폼 안정화 검증을 위해 대량 호 서비스 또한 적용시켜 검증하고 있다. 이러한 서비스들을 동작시켜 보기 위해서는 주변 환경(예, SSP, IP 등)에 대한 구축이 필수적이기에, 이를 위해 간이 SSP, IP등을 개발하였다. 현재의 성능은 연구 수준 정도이지만, 동작 환경의 최적화를 거친다면 많은 성능 향상을 보일 것으로 기대된다.

현재는 플랫폼의 안정성을 높이는 작업을 수행 중이며, 아직까지 개발된 지능망 플랫폼의 최대 성능과 같은 사항은 완전히 연구되지 않았지만, 본 연구에서 갖추어진 환경을 이용하여 새로운 서비스를 개발하여 적용할 수 있을 것으로 예상된다.

향후 연구로는 우선 여러 가지 서비스에 대한 개발과 이에 대한 검증 작업이 수행되어야 할 것이다. 현재 유선망 환경만을 고려하여 구현되었으나, PCS, IMT-2000과 같은 무선망에서도 적용될 수 있는 형태의 지능망 서비스 플랫폼으로 확장하는 연구가 수행될 것이다. 이를 위해, 좀더 다양한 종류의 SIB들을 구현하고, 좀더 편리한 사용자 환경 지원을 계획하고 있다.

참 고 문 헌

[1] Yong Lee, JooSeok Song, An overload control of SCP in advanced intelligent network with fairness and priority, Computer Comm. , Vol.22, NO.2 , 137-143 , 1999

[2] SangChul Song, JiYoung Lee, YeonJoong Kim, SunShin An, Design of SSP Simulator for Wireline/Wireless Intelligent Network, IN2000, May 2000

[3] EURESCOM, Enabling Technologies for IN Evolution and IN-Internet Integration, Project P909-GI, Deliverable 1,2, http://www.eurescom.de

[4] F. Jones, Jambala-Intelligence beyond digital wireless, Ericsson Review No.3, 1998

[5] HP Cisco multiservice controller platform, http://www.communications.hp.com/opencall_

[6] Ramnath A. Lakshmi-Ratan, The Lucent Technologies Softswitch Realizing the Promise of Convergence, Bell Labs Technical Journal, April-June, 1999

[7] Sun Microsystems, JAIN: Integrated Network APIs for the JAVA Platform, http://java.sun.com/products/jain/

[8] S. Beddus, G.Bruce, Opening Up Networks with JAIN Parlay, IEEE Communications Magazine,Vol. 38, Issue 4, April, 2000

[9] ITU-T Rec. Q.1222, Service Plane for Intelligent Network Capability Set-2

[10] ITU-T Rec. Q.1223, Global Functional Plane for Intelligent Network Capability Set-2

[11] ITU-T Rec. Q.1224, Distributed Functional Plane for Intelligent Network Capability Set-2

[12] ITU-T Rec. Q.1225, Physical Plane for Intelligent Network Capability Set-2

[13] 최고봉, 김기령, 김태일, 윤병남, 지능망 기술, 홍릉과학 출판사

이 지 영

1995년 2월 고려대학교 전자공학과(공학사). 1997년 2월 고려대학교 전자공학과(공학석사). 1997년 3월 ~ 현재 고려대학교 전자공학과 박사과정 재학 중. 관심분야는 이동 에이전트, 차세대 지능망, IMT-2000

김 연 중

1995년 2월 고려대학교 전자공학과(공학사). 1997년 2월 고려대학교 전자공학과(공학석사). 1997년 3월 ~ 현재 고려대학교 전자공학과 박사과정 재학 중. 관심분야는 지능망, IMT-2000, VoIP

마 영 식

1994년 2월 아주대학교 공과대학 전자공학과(공학사). 1997년 8월 고려대학교 전자공학과(공학석사). 1999년 8월 고려대학교 전자공학과 박사과정 수료. 관심분야는 컴퓨터 네트워크, Mobile IP, Diff Serv. QoS Routing Algorithm, Ad-hoc Network

김 동 호

1986년 고려대학교 전자공학과(공학사). 1996년 고려대학교 전자공학과(공학석사). 2000년 고려대학교 전자공학과(공학박사). 1985년 12월 ~ 1994년 1월 삼성전자(주) 연구원. 2001년 ~ 현재 한라대학교 정보통신공학부 전임강사. 관심 분야는 고속 통신망 프로토콜, 이동 통신

안 순 신

1973년 서울대학교 공과대학 졸업(B.S). 1975년 한국과학기술원 전기 및 전자과 졸업(M.S). 1979년 블란서 ENSEEIHT에서 공학박사 취득(Ph.D). 1979년 3월 ~ 1982년 8월 아주대학교 전자과 교수. 1991년 1월 ~ 1992년 2월 NIST(National Institute of Standard and Technology) 방문연구원. 1982년 ~ 현재 고려대학교 전자공학과 교수. 관심분야는 컴퓨터 네트워크 및 분산 시스템임