지능망 플랫폼상의 SSP의 설계 및 구현

우시남° 박우진 김연중 이지영 안순신
고려대학교 전자공학과

Design and Implementation of SSP in Intelligent Network Platforms

SiNam Woo°, WooJin Park, YeunJoong Kim, JiYoung Lee and SunShin Ahn
Department of Electronics, Korea University

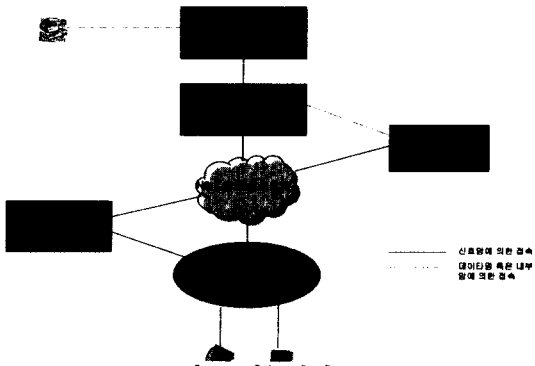
요 약

지능망은 서비스의 독립적인 구현과 표준화된 방법을 제공하고 서비스를 분산적으로 제어할 수 있는 통신망 구조로 오늘날 전기통신의 급속한 기술발전과 신규 전화 가입자의 둔화 ,통신 서비스에 대한 사용자의 요구, 통신 사업자들의 경쟁으로 인하여 지능망 기술에 대한 관심은 커져 가고 있다. 그리고 다양하고 고품질의 서비스를 추구하는 지능망은 통신미디어의 고속화와 광대역화등의 통신매개체의 발달로 인하여 지능망은 더욱 발전하고 있다. 또 전달층과 제어층이 분리되어 있는 지능망을 구축하면 기존망에 영향을 주지 않고 서비스를 도입할 수 있고 안정적으로 서비스를 제공할 수 있다. 지능망 플랫폼을 개발하는데 있어서 호에 대한 처리와 지능망 서비스를 처리하는 Service Switching Point(SSP)의 구현이 필요하다. 이 논문에서는 ITU-T IN CS-2의 SSF/CCF model을 기반으로 해서 SSP에 대한 설계와 구현에 대한 방법을 제시하고 있다.

1.서론

지능망 서비스는 1984년 미국의 AT&T가 분리되면서 설립된 BOC(Bell operting company)사가 독자적인 데이터베이스를 구축하고 800서비스를 제공함으로써 출현하게 되었다. 이러한 지능망 서비스는 다양한 고품질의 서비스를 제공하는 수단으로 발전해 왔다.

지능망은 ITU-T의 주도로 표준화 작업이 이루어지고 있고 TIA/EIA에서도 차세대 지능망(AIN)의 표준화 작업과 WIN(wireless Intelligen Network)의 표준화 작업을 진행되고 있다.

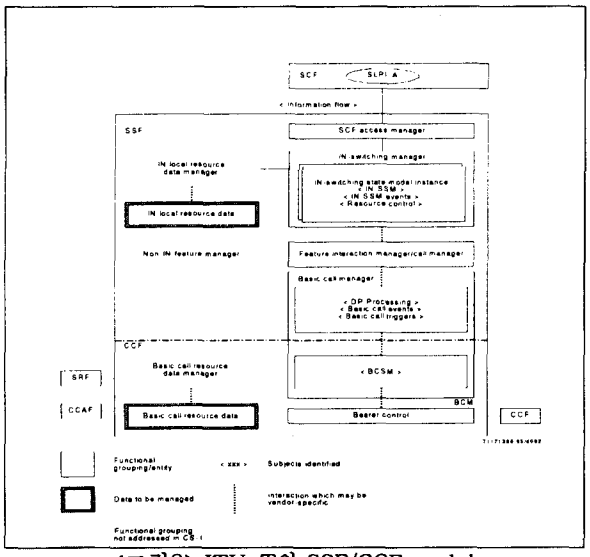


<그림1> 지능망의 구조

지능망의 구조는 그림1과 같다. SCP(Sevice Control Point)는 지능망 서비스 처리에 필요한 서비스 로직 프로그램과 데이터를 가지고 있으며 신호망을 통해서 SSP, IP 등과 연결되어 있다. SSP(Service Switching Point)는 사용자에게 망 접근을 제공하고 지능망 호 연결을 위하여 필요한 교환기능을 제공한다. SDP(Service Data Point)는 지능망 서비스 수행에 필요한 사용자 및 데이터를 가지고 있다. SMP(Service Management Point)는 서비스 관리 제어, 서비스 제공 제어 그리고 서비스 전개 제어를 수행한다. IP(Intelligent Peripheral)는 사용자와 망간의 융통성 있는 정보의 상호작용을 위하여 특수 자원을 관리한다. 본 논문의 SSP설계는 ITU-T의 IN CS-2(intelligent network capability set-2)에 기반을 두어서 설계를 했다. 2장에서는 SSP의 기능과 구조, 3장에서는 SSP의 설계와 구현, 마지막으로 4장에서는 결론을 내리고 있다.

2.SSP의 기능과 구조

SSP는 사용자에게 망 접근을 제공하거나(SSP가 local인 경우) 교환 기능을 수행하는 능력에 추가적으로 SSP는 지능망의 관련된 서비스에 필요한 교환기능을 제공한다. 이를 위하여 지능망 서비스 요청을 감지하기 위한 감지능력, SCF를 포함하는 SCP와 같은 물리 실체와 통신하는 통신 능력 및 다른 물리실체로부터 전송되어 오는 명령에 대한 응답능력을 포함한다. SSP의 기능은 그림과 같이 CCF, SSF를 포함하며 선택적으로 SCF, SRF, SDF, CCAF를포함할 수 있다.



<그림2> ITU-T의 SSF/CCF model

본 논문은 IN CS-2에 나와 있는 SSF/CCF 모델를 기반으로 해서 SSP 설계를 했다.

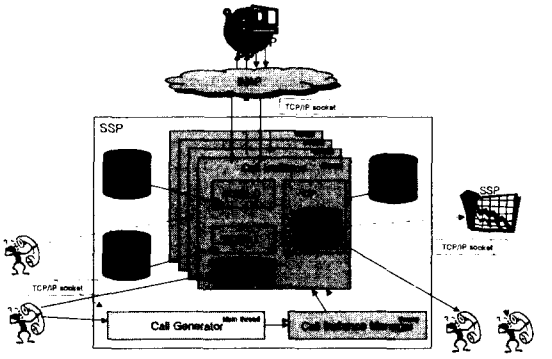
SSF/CCF model은 BCM(basic call manageri), IN-SM(IN-switching manager), FIM/CM(feature interations manager/call manager), 각 구성요들간의 관계로 구성되어 있다. BCM은 사용자를 위해서 연결 통로를 만들어 주는 기본호와 연결 제어를 위한 추상적인 부분이고, 기본호와 연결 제어에 필요한 자원을 관리하고 서비스 로직 인스턴스를 발생시키는 기본호 또는 연결 제어 사건을 감지한다. 이것은 BCSM고 DP processing으로 이루어 진다. IN-SM는 지능망 서비스를 사용자에게 제공하기 위해서 SCP와 상호작용하는 기능 실체이다. 그것은 SSF/CCF에서의 호/연결 과정을 제공하고, SSF/CCF의 자원에 접근할 수 있으며 지능망 서비스를 나타내는 사건을 감지하고, 그리고 지능망 서비스 로직 인스턴스를 지원하는데 필요한 SSF의 자원을 관리한다. FIM/CM는 하나의 호에 대해서 지능망 서비스 로직 인스턴스와 비지능망 서비스 인스턴스를 동시에 제공할 수 있는 기능 실체이다.

BCSM은 지능망 서비스로직 인스턴스가 기본호, 연결 제어 Capabilities와 상호작용할 때 호 처리과정를 나타낸다. 특히 지능망 서비스 로직 인스턴스를 요구하거나 또는 활성화된 지능망 서비스 로직 인스턴스에 알려주어야 하는 사건을 명시하고 있다. 다시 말하면 호 처리 과정과 사건이 감지되었을 때의 행동을 명시하고 있다. BCSM은PIC(points in Call), DP(detection Point), BCSM transition, events로 이루어진다.

3. SSP 설계 및 구현

설계한 SSP는 IN CS-2에 기반으로 해서 지능망 플랫폼 구현에 있어서 호제어, 지능망 비스 처리기능만을 가지고 있다. 본 논문의 SSP의 구조는 그림3과 같다.

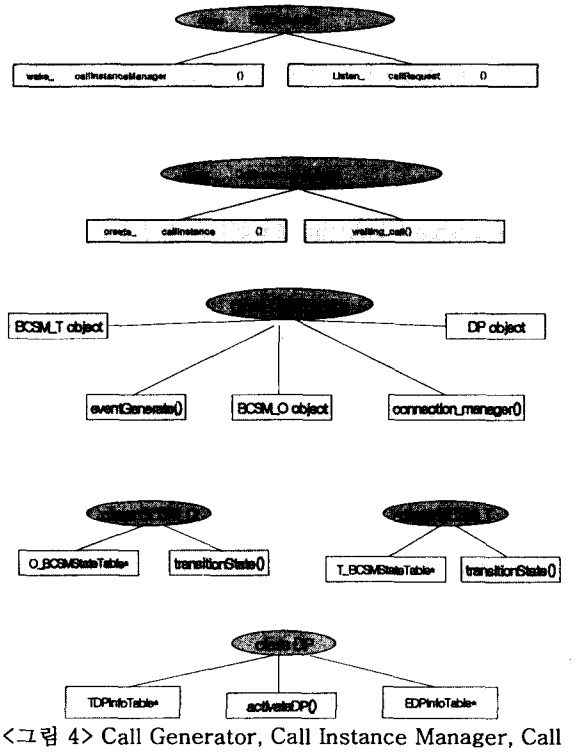
SSP의 구조는 하나의 프로세스에서 구현이 이루어졌다. 그리고 Posix 쓰레드를 사용했다. SSP는 크게 Call Generator ,Call Instance Manager, Call Instance로 구성되어 있다. 이 구성 요소들은 하나의 쓰레드로 존재한다. 다른 요소간의 통신은 TCP/IP socket을 통해서 이루어 졌고 SSP안의 각 쓰레드간의 통신은 ThreadIPC(Thread Interprocessing Communication)를 통해서 이루어졌다.



<그림3>SSP의 구조

Call Generator

*Call Generator*는 사용자로부터 호 요청을 받고 이 메세지를 Call Instance Manager에 전달하는 기능을 가지고 있는 클래스이다. 여기서 Call Instance Manager에 메시지를 보내는 방법은 Posix에 있는 pthread_cond_signal()을 통해서 신호을 발생시키는 것이다. 그리고 Call Generator는 다른 사용자로부터 또 호 요청을 받기 위해 Accept (TCP socket function)상태에 들어가게 된다.



<그림 4> Call Generator, Call Instance Manager, Call Instance의 구조

Call Instance Manager

Call Instance Manager는 하나의 호마다 Call Instance를 생성하는 역할을 하는 클래스이다. Call Instance를 생성하자마자 Call Instance Manager는 pthread_cond_wait()를 부르고 블럭상태에 들어가고 Call Generator의 pthread_cond_signal()에 의해 발생되는 신호을 기다린다. 다시 신호을 받으면 다시 Call Instance를 탄생시키고 또 다시 블럭된다.

Call Instance

Call Instance는 호를 처리하는 주된 역할을 수행하는 클래스로써 호마다 쓰레드가 생성된다. Call Instance의 구조는 그림4와 같다. 하부구조에 발생하는 외부와의 연결을 요청하는 경우 Call Instance 가 모두 처리하는 루틴을 가지고 있다.(예를 들면 SCP와 통신, 사용자와의 통신, 다른 SSP와의 통신이 있다.) 이런 구조로 인하여 각각의 쓰레드가 독립적으로 호처리를 하게 된다. 그리고 외부와의 통신에 생기는 문제를 일괄적으로 처리할 수 있다. Call Instance는 BCSM_O, BCSM_T, DP로 구성되어 있다.

BCSM_O - 발신 BCSM의 역할을 수행하는 클래스이고 발신 BCSM의 상태를 가지는 테이블에 대한 포인터를 가지고 있다. 이 테이블를 참조해서 BCSM_O의 상태를 바꾸어 준다.

BCSM_T - 착신 BCSM_T의 역할을 수행하는 클래스이고 착신 BCSM의 상태를 가지는 테이블에 대한 포인터를 가지고 있다. 이 테이블을 참조해서 BCSM_T의 상태를 바꾸어 준다.

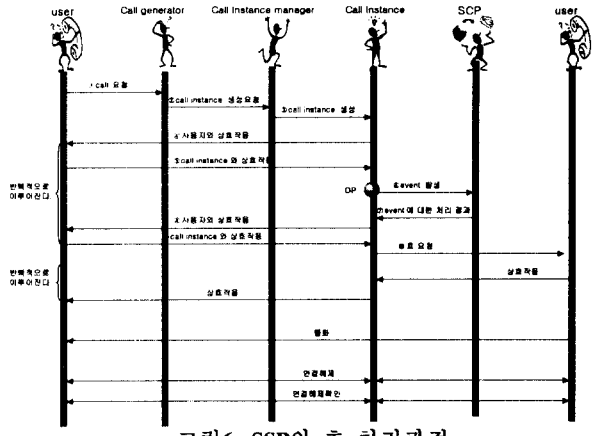
DP - BCSM의 상태가 DP에 있을 때 DP의 행동을 수행하는 클래스이다. DP는 TDPInfoTable과 EDPInfoTable에 대한 포인터를 가지고 있고 이 테이블을 참조의해서 DP의 적절한 행동을 결정한다. TDPTable은 SSP에 하나만 있지만 EDPTable은 각 호마다 존재한다.

```
class O_BCSMStateTable{
current state          //현재 상태를 표시한다.
Event*                 //class Event 의 object 이다. Single linked list 로 구현되어 있다.
};        //T_BCSMStateTable 도 같은 구조를 가지고 있다.

class Event{
next state:            //다음 상태를 표시한다.
Event:                 //특정 event 를 발생시킨다.
Event* next:           //linked list 로 구현이 되어 있기 때문에 다음에 대한 포인터이다.
};

class TDPInfoTable{
DPNumber :             //DPNumber
ArmedState :           //DP 가 armed 되어 있는 상태를 나타낸다.
Criteria:              //class Criterial 의 object 이다.
};

class  Criteria{
Trigger asigned :      //트리거의 상태를 나타내며 cond 와 uncond 가 있다.
specific digits:       //다음 3가지는 구조체로 이루어져 있다. 이 구조체는 digits, seviceID.
callingPartyNumber :   // next pointer 로 구성되어 있다. Linked list 로 구현되어있다.
calledParytNumber :
};
```

<그림5> Call Instance 하부 구조에 대한 코드

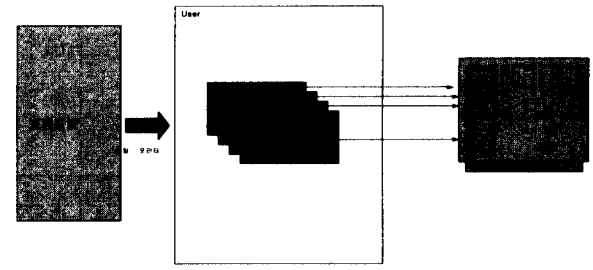
그림6은 본 논문에서 SSP의 호 처리과정을 나타낸 것이다.

본 논문의 SSP는 지능망 플랫폼을 테스트를 하기 위한 것이기 때문에 대량호에 대한 처리과정도 필요하다. 대량호를 처리하기 위해 호에 대한 정보를 파일을 통해서 받는다. 그리고 그 파일을 읽어서 하나의 호마다 thread를 탄생시켜서 대량호를 처리하게 한다.



<그림6> SSP의 호 처리과정

즉 사용자와 SSP가 통신을 할 때에 프로세스의 각각의 쓰레드간에 통신이 이루어지는 것이다. 처리과정은 그림7과 같다.



<그림 7> 사용자측의 대량호 처리과정

4.결론

본 논문에서 제시한 SSP는 ITU-T의 CCF/SSF 모델에 기반으로 설계하였다. 그리고 본 논문은 지능망 플랫폼을 시뮬레이션하는데 필요한 SSP를 설계한 것이기 때문에 지능망 플랫폼을 실제 물리 실체에 도입하기에 앞서 검증할 수 있는 수단을 제시하고 있다. 앞으로 객체 지향적이고 호 처리에 대한 중앙처리 방식의 설계가 필요하며, 또 다른 물리 실체간의 인터페이스의 개선이 필요할 것이다.

참고 문헌

[1]ITU-T Rec. Q.1224, "Distributed Functional Plane for Intelligent Network CS-2", 1997.
[2]ITU-T Rec. Q.1220 ~ Q.1223, Q.1225, "Intelligent Network capability Set 2", 1997.
[3]ITU-T Rec. Q.1228, "Interface Recommendation for Intelligent Network Capability Set", 1997.
[4]Scott J.Norton, Mark D.Disasquale, "THREADTIME", 1997.
[5]David R. Butenhof, "Programming with POSIX Threads" 1997.
[6]최고봉, 김기령, 김태일, 윤병남, 홍릉 과학 출판사, "지능망 기술", 1996.
[7]송상철, 김연중, 이지영, 안순신, "지능망 플랫폼 시뮬레이션을 위한 간이 SSP 설계" 1999.