



(19)대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。 Int. Cl.		(45) 공고일자	2007년08월03일
H04L 12/28 (2006.01)		(11) 등록번호	10-0746281
H04L 12/24 (2006.01)		(24) 등록일자	2007년07월30일
(21) 출원번호	10-2006-0077186	(65) 공개번호	
(22) 출원일자	2006년08월16일	(43) 공개일자	
심사청구일자	2006년08월16일		

(73) 특허권자	서울통신기술 주식회사 서울 강동구 성내3동 448-11
(72) 발명자	한명희 경기 수원시 영통구 원천동 327 CS아카데미 7층
(74) 대리인	박상수
(56) 선행기술조사문헌	KR1020020005802 A

심사관 : 김대성

전체 청구항 수 : 총 7 항

(54) 공유 메모리를 이용한 S N M P 네트워크 통신 시스템 및통신 방법

(57) 요약

본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 시스템은 공유 메모리를 생성하고, 생성된 상기 공유 메모리에 시스템의 상태 변화나 성능/통계 값을 저장하는 OLT 시스템 데몬; 및 외부 네트워크 관리 시스템으로부터 데이터 get 요청 받으면, 상기 공유 메모리에 접속하여 해당 데이터를 읽어(read)들인 후 데이터 get 메시지를 처리하여 외부 네트워크 관리 시스템으로 전송하는 SNMP 에이전트를 포함하는 것으로, Net_SNMP 에이전트와 연동 데몬(Daemon)간에 요청/응답 절차가 없어짐에 따라 지연 요소를 제거하여 시스템의 안전성을 보장받을 수 있는 것이다.

대표도

도 2

특허청구의 범위

청구항 1.

SNMP 네트워크 시스템에 있어서,

공유 메모리를 생성하고, 생성된 상기 공유 메모리에 시스템의 상태 변화나 성능/통계 값을 저장하는 OLT 시스템 데몬; 및 외부 네트워크 관리 시스템으로부터 데이터 get 요청 받으면, 상기 공유 메모리에 접속하여 해당 데이터를 읽어(read)들인 후 데이터 get 메시지를 처리하여 외부 네트워크 관리 시스템으로 전송하는 SNMP 에이전트를 포함하는 공유 메모리를 이용한 SNMP 네트워크 통신 시스템.

청구항 2.

제 1항에 있어서,

상기 OLT 시스템 데몬은,

상기 공유 메모리 생성 시, 자신의 내부 메모리와의 연결을 위해 상기 공유 메모리의 지정 키, 공유 메모리의 세그먼트 크기, 및 공유 메모리에 대한 접근 권한 설정을 위한 플래그를 설정하는 것을 특징으로 하는 공유 메모리를 이용한 SNMP 네트워크 통신 시스템.

청구항 3.

제 2항에 있어서,

상기 플래그는,

공유 메모리와 연결되는 프로세스의 내부 어드레스를 시스템이 관리하도록 연결 영역을 대체하는 SHM_RND와, 읽기 전용으로 메모리 연결을 지정하는 SHM_RDONLY, 및 연결 영역을 대체하는 SHM_REMAP인 것을 특징으로 하는 공유 메모리를 이용한 SNMP 네트워크 통신 시스템.

청구항 4.

제 1항에 있어서,

상기 공유 메모리는,

물리적인 메모리를 이용하는 것을 특징으로 하는 공유 메모리를 이용한 SNMP 네트워크 통신 시스템.

청구항 5.

OLT 시스템 데몬을 통해 공유 메모리를 생성하는 단계;

상기 OLT 시스템 데몬이 시스템의 상태 변화나 성능/통계 값을 상기 공유 메모리에 저장하는 단계; 및

외부 네트워크 관리 시스템으로 get 요청을 받으면, SNMP 에이전트가 생성된 공유 메모리에 접속하여 해당 데이터를 읽어들인 후 외부 네트워크 관리 시스템으로 해당 데이터의 get 처리를 수행하는 단계를 포함하는 SNMP 시스템에서 공유 메모리를 이용한 통신 방법.

청구항 6.

제 5항에 있어서,

상기 OLT 시스템 데몬을 통해 공유 메모리를 생성하는 단계는,

상기 OLT 시스템 데몬의 내부 메모리와의 연결을 위해 상기 공유 메모리의 지정 키, 공유 메모리의 세그먼트 크기, 및 공유 메모리에 대한 접근 권한 설정을 위한 플래그를 설정하는 것을 특징으로 SNMP 시스템에서 공유 메모리를 이용한 통신 방법.

청구항 7.

제 6항에 있어서,

상기 플래그는,

공유 메모리와 연결되는 프로세스의 내부 어드레스를 시스템이 관리하도록 연결 영역을 대체하는 SHM_RND와, 읽기 전용으로 메모리 연결을 지정하는 SHM_RDONLY, 및 연결 영역을 대체하는 SHM_REMAP인 것을 특징으로 하는 공유 메모리를 이용한 SNMP 시스템에서 공유 메모리를 이용한 통신 방법.

명세서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 공유 메모리를 이용한 SNMP 네트워크 통신 시스템 및 통신 방법에 관한 것이다.

종래 SNMP 에이전트로 임의의 데이터에 대한 get이 이루어질 경우 Net_SNMP 에이전트는 IPC나 TCP/IP 프로토콜 방식을 이용하여 메시지 형태로 요청하고 메시지 형태로 응답하는 방법을 이용하였다.

즉, 도 1에 도시된 바와 같이 네트워크 관리 시스템(1)으로부터 get 요청을 SNMP 에이전트(10)가 받게 되면, 상기 SNMP 에이전트(10)는 Net_SNMP는 IPC나 TCP/IP를 이용하여 해당 OLT 시스템 데몬(20)에 get 요청을 하게 된다.

상기 SNMP 에이전트(10)로부터 get 요청을 받은 해당 OLT 시스템 데몬(20)은 get 요청에 해당하는 데이터를 상기 SNMP 에이전트(10)로 전송한다.

이어서, 상기 OLT 시스템 데몬(20)으로부터 get 요청에 따른 해당 데이터를 수신한 SNMP 에이전트(10)는 get 처리하여 상기 네트워크 관리 시스템(1)으로 응답하도록 되어 있다.

그러나 종래 SNMP 네트워크 시스템은 SNMP 에이전트의 get 요청으로부터 상기 OLT 시스템 데몬(20)의 응답이 바로 오지 않고 지연이 발생할 경우 SNMP 에이전트(10)는 hanging 또는 장애가 발생할 수 있는 문제점을 가지고 있었다.

그 뿐만 아니라, 통계나 성능 데이터의 경우 그 수가 많기 때문에 그때그때 요청과 응답을 반복하게 되면 시스템 내에 IPC가 증가되어 시스템의 성능을 저하할 수 있는 문제점을 가지고 있었다.

발명이 이루고자 하는 기술적 과제

따라서 본 발명은 상기와 같은 종래의 문제점을 해결하기 위한 것으로서, 본 발명의 목적은 SNMP 에이전트로 데이터 get에 대한 요청이 있을 경우 응답 지연을 없애고 에이전트의 안정화를 보장할 뿐만 아니라, 통계나 성능 데이터 같이 트래픽이 많은 데이터의 경우에도 시스템 내부 IPC의 로드를 없애 시스템의 성능을 크게 향상시키는 공유 메모리를 이용한 SNMP 네트워크 통신 시스템을 제공하는 데 있다.

발명의 구성

상기한 목적을 달성하기 위한 본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 시스템의 일 측면에 따르면, 공유 메모리를 생성하고, 생성된 상기 공유 메모리에 시스템의 상태 변화나 성능/통계 값을 저장하는 OLT 시스템 데몬; 및 외부 네트워크 관리 시스템으로부터 데이터 get 요청 받으면, 상기 공유 메모리에 접속하여 해당 데이터를 읽어(read)들인 후 데이터 get 메시지를 처리하여 외부 네트워크 관리 시스템으로 전송하는 SNMP 에이전트를 포함한다.

여기서, 상기 OLT 시스템 데몬은, 상기 공유 메모리 생성 시, 자신의 내부 메모리와의 연결을 위해 상기 공유 메모리의 지정 키, 공유 메모리의 세그먼트 크기, 및 공유 메모리에 대한 접근 권한 설정을 위한 플래그를 설정하는 것을 특징으로 하며, 상기 플래그는 공유 메모리와 연결되는 프로세스의 내부 어드레스를 시스템이 관리하도록 연결 영역을 대체하는 SHM_RND와, 읽기 전용으로 메모리 연결을 지정하는 SHM_RDONLY, 및 연결 영역을 대체하는 SHM_REMAP인 것을 특징으로 한다.

본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 시스템에서의 통신 방법의 일 측면에 따르면, OLT 시스템 데몬을 통해 공유 메모리를 생성하는 단계; 상기 OLT 시스템 데몬이 시스템의 상태 변화나 성능/통계 값을 상기 공유 메모리에 저장하는 단계; 및 외부 네트워크 관리 시스템으로 get 요청을 받으면, SNMP 에이전트가 생성된 공유 메모리에 접속하여 해당 데이터를 읽어들인 후 외부 네트워크 관리 시스템으로 해당 데이터의 get 처리를 수행하는 단계를 포함한다.

여기서, 상기 OLT 시스템 데몬을 통해 공유 메모리를 생성하는 단계는, 자신의 내부 메모리와의 연결을 위해 상기 공유 메모리의 지정 키, 공유 메모리의 세그먼트 크기, 및 공유 메모리에 대한 접근 권한 설정을 위한 플래그를 설정하는 것을 특징으로 하며, 상기 플래그는 공유 메모리와 연결되는 프로세스의 내부 어드레스를 시스템이 관리하도록 연결 영역을 대체하는 SHM_RND와, 읽기 전용으로 메모리 연결을 지정하는 SHM_RDONLY, 및 연결 영역을 대체하는 SHM_REMAP인 것을 특징으로 한다.

이하, 본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 시스템에 대한 바람직한 실시예에 대하여 첨부한 도면을 참조하여 상세하게 살펴보기로 한다. 이 때, 아래에서 설명하는 시스템 구성은 본 발명의 설명을 위해서 인용한 시스템으로써 아래 시스템으로 본 발명을 한정하지 않음을 이 분야의 통상의 지식을 가진 자라면 이해해야 할 것이다.

도 2는 본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 시스템의 구성을 나타낸 도면으로서, 본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 시스템은 공유 메모리(130)를 포함하는 OLT 시스템 데몬(110)과 SNMP 에이전트(120)를 포함한다.

상기 OLT 시스템 데몬(110)은 공유 메모리(130)를 생성하고, 생성된 상기 공유 메모리(130)에 시스템의 상태 변화나 성능/통계 값을 저장한다. 이 때, 상기 OLT 시스템 데몬(110)은 상기 공유 메모리(130) 생성 시, 자신의 내부 메모리와의 연결을 위해 상기 공유 메모리(130)의 지정 키, 공유 메모리(130)의 세그먼트 크기, 및 공유 메모리(130)에 대한 접근 권한 설정을 위한 플래그를 설정하며, 상기 플래그는 공유 메모리(130)와 연결되는 프로세스의 내부 어드레스를 시스템이 관리하도록 연결 영역을 대체하는 SHM_RND와, 읽기 전용으로 메모리 연결을 지정하는 SHM_RDONLY, 및 연결 영역을 대체하는 SHM_REMAP으로 설정할 수 있다.

그리고 상기 SNMP 에이전트(120)는 외부 네트워크 관리 시스템(1)으로부터 데이터 get 요청을 받으면, 상기 공유 메모리(130)에 접속하여 해당 데이터를 읽어(read)들인 후 데이터 get 메시지를 처리하여 외부 네트워크 관리 시스템(1)으로 전송한다.

상기한 각 구성들에 대한 일반적인 기능 및 각각의 상세한 동작에 대하여는 그 설명을 생략하고, 본 발명에 상응하는 동작 위주로 그 동작들을 설명하기로 한다.

먼저, SNMP 네트워크에서 공유 메모리(130)를 설정한다.

여기서, 공유 메모리(130)를 설정하는 방법은 #shmget을 이용한다. 즉, shmat() 시스템 호출을 이용하여 물리적인 메모리에 공유 메모리(130)를 생성하게 되거나, 기존에 생성되어 있는 공유 메모리(130) 정보를 얻어와 shmat() 시스템 호출을 사용하여 프로세스 내부에 있는 가상 메모리와 연결시켜 준다.

상기 shmat() 함수의 프로토콜 타입은 다음과 같다.

```
void *shmat(int snmp, const void *shm_addr, int flag):
```

그리고 상기 shmget() 함수가 사용하는 첫 번째 인수인 키(key)는 여러 개의 프로세스들이 공통의 공유 메모리(130)를 사용할 수 있도록 지정하는 키 값이 되며, 두 번째 인수인 크기(size)는 공유 메모리(130)의 세그먼트 크기가 된다.

마지막 인수로 사용되는 shmflg는 공유 메모리(130)에 대한 접근 권한을 위한 플래그들과 " | " 기호를 이용하여 or 연산을 수행할 수 있다. 만약, IPC_CREAT 플래그와 함께 세팅하게 되면 공유 메모리(130)가 생성되어 있지 않을 때 새로 생성되도록 만들 수 있다.

이 때, 이미 생성된 공유 메모리(130)가 있다면 IPC_CREAT에 대한 설정은 무시가 되고 이미 만들어 있던 공유 메모리(130)로 리턴 된다. 그리고 shmget()함수는 실행에 실패하게 되어 -1을 리턴하게 된다.

shmget() 함수의 호출로 생성되는 공유 메모리(130) 영역은 프로세서의 가상 메모리 영역이 아닌 실제 메모리의 물리적 영역이 된다. 따라서 shmget() 시스템 호출의 간단한 사용 예를 보도록 한다.

```
#include <sys/types.h>
```

```
#include <sys/ipc.h>
```

```
#include <sys/shm.h>
```

```
#define COMMANDSIZE 64 /공유 메모리로 사용할 크기를 선언/
```

```
int smld = shmget((key_t)9000, COMMANDSIZE, 0666 | IPC_CREAT); /shmget을 이용하여 공유 메모리를 확보/
```

```
if(smld == -1) {
```

```
printf("shmget 실행 실패 Wn);
```

```
exit(0);
```

```
}
```

상기와 같이 공유 메모리(130)를 설정한다.

이후, SNMP 에이전트(120)는 shmget() 시스템 호출을 이용하여 물리적인 메모리에 생성된 공유 메모리(130) 또는 기존에 생성되어 있던 공유 메모리(130)의 정보를 얻어왔다면, shmat() 시스템 호출을 사용하여 프로세스 내부에 있는 가상 메모리와 연결시켜 주어야 한다. 이 때, 상기 shmat() 함수의 프로토콜 타입은 다음과 같다.

```
void *shmat(int smld, const void *shm_addr, int flag);
```

shmat() 함수를 이용하여 가상 메모리와 메모리의 물리적 영역을 연결한 다음 공유 메모리(130)로 활용할 수 있다.

따라서 shmat() 함수의 첫 번째 인수인 smld는 shmat() 함수 호출을 통해 얻어온 공유 메모리(130)의 구술어가 된다.

두 번째 인수인 shm_addr는 공유 메모리(130)와 연결하고자 하는 프로세스 내부의 메모리를 가리킨다. 그러나 프로세스 내부의 메모리 영역을 개발자가 지정 해주는 것 보다 시스템이 상황에 맞게 지정하도록 만들어 주는 것이 좋기 때문에 일반적으로 0으로 세팅하게 된다.

마지막 인수로 사용되는 flag는 공유 메모리(130)와 프로세스 내부의 메모리를 연결하면서 원하는 속성을 지정하는데 사용되는 플래그로 or 연산을 통한 비트 세팅이 가능하다. 이 때, 사용될 수 있는 선언자로는 공유 메모리(130)와 연결되는 프로세스의 내부 어드레스를 시스템이 관리하도록 연결 영역을 대체하는 SHM_RND와 읽기 전용으로 메모리 연결을 지정하는 SHM_RDONLY, 그리고 연결 영역을 대체하는 SHM_REMAP 등으로 설정한다.

이러한, shmat() 함수는 공유 메모리(130)와 연결이 완료된 프로세스 내부의 가상 메모리 주소를 리턴해 준다. 만약 실패하면 -1을 리턴하게 된다.

그럼, shmat() 함수를 이용하여 메모리 주소를 얻어내고 이를 활용하기 위해 프로그램 내부의 버퍼와 연결하는 간단한 방법을 살펴본다.

```
int smld = shmget((key_t)9000, COMMANDSIZE, 0666 | IPC_CREAT); /shmget() 함수를 이용하여 공유 메모리의 ID를 구함/
```

```
s_memory = shmat(smld, (void *)0, 0);
```

```
if(s_memory == (void *) -1) {
```

```
printf("shmat 실행실패 %n");
```

```
exit(0);
```

```
}
```

```
#int epon_sm_init()는
```

공유 메모리(130) 생성은 다른 프로세스 데몬에서 생성하고, SNMP 프로세스는 생성된 공유 메모리(130)를 공유한다.

만약, 공유 메모리(130)가 생성되어 있지 않으면 SNMP 프로세스는 실행되지 않는다.

```
/* 공유 메모리의 KEY, SIZE, ID */
```

```
int SNMP_smKEY = 0;
```

```
int SNMP_smSIZE = 0;
```

```
int SNMP_smID = 0;
```

```
/* 공유 메모리 속의 데이터와 시작 주소를 참조할 포인터 */
```

```
void *snmp_memory = NULL;
```

```
int epon_sm_init()
```

```
{
```

```
SNMP_smKEY = KEY_eponOltDeviceInfo_DB;
```

```
SNMP_smSIZE = SIZE_eponOltDeviceInfo_DB;
```

```
/* shmget를 이용하여 공유 메모리를 확보 */
```

```
snmp_smID = shmat((key_t)snmp_smKEY, snmp_smSIZE, SHM_RDONLY);
```

```
if(snmp_smID == -1)
```

```
snmp_log(LOG_INFO, "ROZEN : epon_system_sm_init1 %n");
```

```
return 0;

snmp_memory = shmat(snmp_smlD, (void *)0, 0);

if(snmp_memory == (void *) -1)

snmp_log(LOG_INFO, "ROZEN : epon_system_sm_init2Wn");

return 0;

/* 공유 메모리 주소와 내부 변수 포인터 연결 */
```

eponOltDeviceInfo_DB = (eponOltDeviceInfo_info *)snmp_memory; 와 같이 구현된다.

그러면, 상기와 같은 구성을 가지는 본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 시스템에서의 통신 방법에 대해 도 3을 참조하여 설명하기로 한다.

먼저, OLT 시스템 데몬(110)을 통해 공유 메모리(130)를 생성한다(S1). 여기서, 상기 OLT 시스템 데몬(110)을 통해 공유 메모리(130)를 생성하는 단계는, 자신의 내부 메모리와의 연결을 위해 상기 공유 메모리(130)의 지정 키, 공유 메모리(130)의 세그먼트 크기, 및 공유 메모리(130)에 대한 접근 권한 설정을 위한 플래그를 설정하는 것을 특징으로 하며, 상기 플래그는 공유 메모리(130)와 연결되는 프로세스의 내부 어드레스를 시스템이 관리하도록 연결 영역을 대체하는 SHM_RND와, 읽기 전용으로 메모리 연결을 지정하는 SHM_RDONLY, 및 연결 영역을 대체하는 SHM_REMAP인 것을 특징으로 한다.

이어서, 상기 OLT 시스템 데몬(110)이 시스템의 상태 변화나 성능/통계 값을 상기 공유 메모리(130)에 저장한다(S2).

이 후, 외부 네트워크 관리 시스템(1)으로 get 요청을 받으면, SNMP 에이전트(120)가 생성된 공유 메모리(130)에 접속하여 해당 데이터를 읽어들이고 후 외부 네트워크 관리 시스템(1)으로 해당 데이터의 get 처리를 수행한다(S3).

이상에서 본 발명은 기재된 구체적인 실시예에 대해서만 상세히 설명하였지만 본 발명의 기술 사상 범위 내에서 다양한 변형 및 수정이 가능함은 당업자에게 있어서 명백한 것이며, 이러한 변형 및 수정이 첨부된 특허청구범위에 속함은 당연한 것이다.

발명의 효과

상술한 바와 같이 본 발명에 의한 공유 메모리를 이용한 SNMP 네트워크 통신 시스템에 의하면, Net_SNMP 에이전트와 연동 데몬(Daemon)간에 요청/응답 절차가 없어짐에 따라 지연 요소를 제거하여 시스템의 안전성을 보장받을 수 있는 뛰어난 효과가 있습니다.

또한, Net_SNMP 에이전트와 시스템 데몬간에 IPC를 이용하지 않기 때문에 내부 IPC 트래픽을 혁신적으로 줄일 수 있으며, Net_SNMP와 연동 데몬간에 요청/응답 절차가 없어짐에 따라 빠른 응답성을 가지며 CPU 로드를 감소시킬 수 있는 뛰어난 효과가 있습니다.

도면의 간단한 설명

도 1은 종래 SNMP 네트워크 통신 시스템의 구성을 나타낸 기능블록도.

도 2는 본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 시스템의 구성을 나타낸 기능블록도.

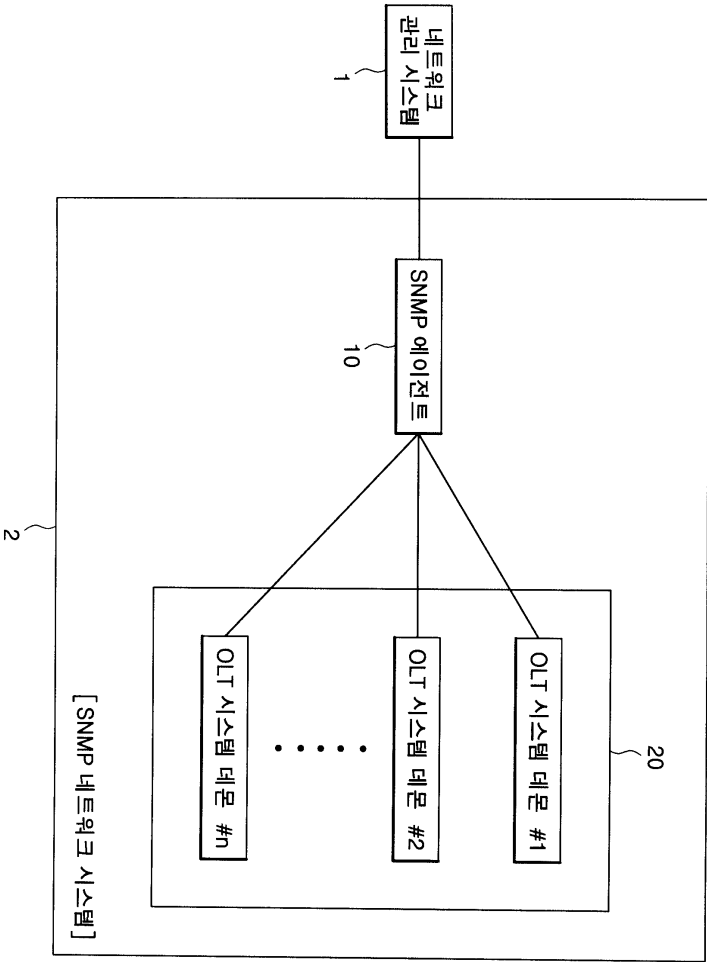
도 3은 본 발명에 따른 공유 메모리를 이용한 SNMP 네트워크 통신 방법을 나타낸 순서도이다.

<도면의 주요 부분에 대한 부호의 설명>

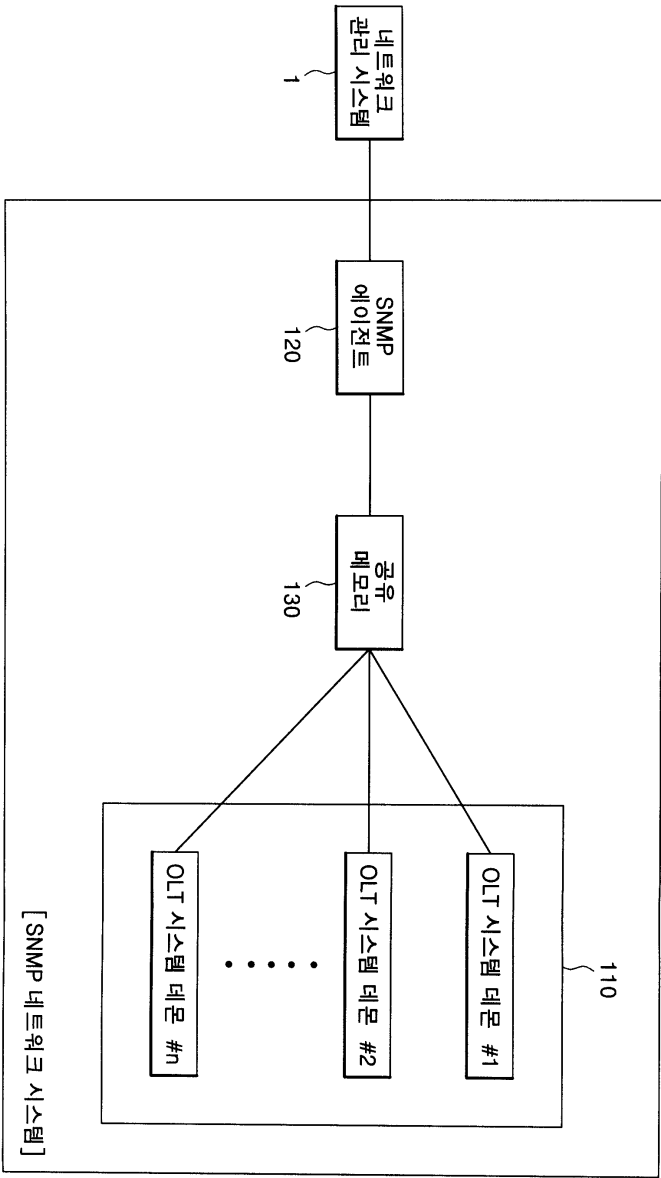
1 : 네트워크 관리 시스템 110 : OLT 시스템 데몬
120 : SNMP 에이전트 130 : 공유 메모리

도면

도면1



도면2



도면3

