DB 및 리눅스 FUSE 기반 IoT 서비스 플랫폼을 위한 가상 파일시스템 구현

이형봉*
*강릉원주대학교 컴퓨터공학과
e-mail:hblee@gwnu.ac.kr

Implementation of a DB-based Virtual Filesystem for IoT Service Platform using Linux FUSE

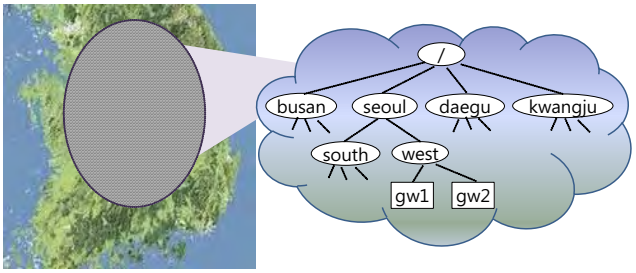
Hyung-Bong Lee*
*Dept of Computer Science & Engineering, Gangneung-Wonju National University

요 약

IoT의 주요 구성요소는 기존의 데스크 탑 외에 디바이스 즉, 단말 기기들이 주류를 이룬다. 이러한 IoT 디바이스들은 데이터의 유형이나 접근 방법이 다양하고, 실시간적 데이터 생산과 제어를 위한 양방향 데이터 접근 지원을 필요로 한다. 이러한 IoT 디바이스를 연결하여 클라우드 형 서비스로 개발하기 위해서는 디바이스 속성 관리가 용이한 도메인 관리 방법과 디바이스에 대한 일관된 접근 인터페이스를 제공하는 플랫폼이 필요하다. 이 논문에서는 리눅스 파일시스템 후면 즉, 사용자 영역에 리눅스 파일 시스템 스타일의 DB 기반 가상 파일시스템을 구축하여 IoT 디바이스를 연결하고 관리하는 프레임워크를 제시한다. 구현 가상 파일시스템은 계층적 디렉터리 체계를 DB에 유지하면서 단말 노드에는 지리적으로 산재한 IoT 디바이스들에 대한 속성 정보를 관리한다. 디렉터리 및 IoT 디바이스의 추가・삭제・검색 등 도메인 관리는 mkdir. mknod, ls, find 등 리눅스 고유 명령어로 이루어지고, 모든 IoT 디바이스에 대한 접근은 open(), read(), write(), close() 등 POSIX 인터페이스를 통해 가능하다.

1. 서론

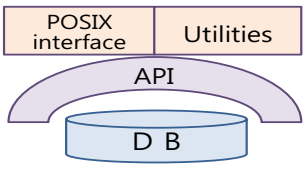
IoT의 주요 구성요소는 센서 등 디바이스들이고, 이들 센서들이 생산하는 실시간 데이터를 활용하여 개발할 수 있는 서비스는 무궁무진하다. 이와 같이 IoT 데이터를 활용하고자 하는 기업이나 개인은 지리적으로 산재해 있는 디바이스가 통합적으로 관리되고 이들에 대한 편리한 접근 인터페이스가 제공되는 개발 플랫폼 즉, 작업장을 필요로 한다. 이 논문에서는 그림 1과 같이 전국에 흩어져 있는 디바이스들을 리눅스 파일시스템 뷰로 통합하여 관리하는 DB 기반 가상 파일시스템을 제시한다. 이 가상 파일시스템에서는 mkdir, rmdir 등 리눅스 고유 명령어에 의해 디렉터리 체계가 운영되고, IoT 디바이스와 대응되는 노드는 mknod, rm, ls, find 등 리눅스 고유 명령어에 의해 추가・삭제・검색 등이 이루어진다.



(그림 1) IoT 서비스 플랫폼을 위한 가상 파일시스템 개념

2. 관련연구

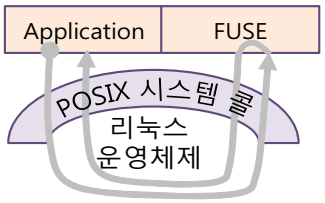
그림 1의 뷰 관리를 위한 방법의 하나로 그림 2와 같이 유닉스 파일시스템을 가상으로 관리하는 DB 위에 DB 접근 API를 두고, POSIX 스타일의 인터페이스와 유틸리티를 제공하는 방안이 제안되었다[1]. 이 제안에서는 가상 파일시스템 접근 API와 open() 등 POSIX 인터페이스, 그리고 sh, mkdir 등 유틸리티들이 모두 사용자 영역에서 새롭게 구현되어야 한다. 이로 인해 사용자 관리, 입・출력 모드, 보안 모드, 프로세스 관리 등의 측면에서 POSIX와 완벽하게 호환된 인터페이스 제공에 한계가 있고, 무엇보다도 사용자들은 운영체제가 제공하는 고유의 시스템 콜을 사용할 수 없다는 단점이 있다. 이 연구에서는 리눅스 FUSE(Filesystem in Userspace)를 이용하여 DB 기반 가상 파일시스템을 리눅스 파일시스템 아래에 배치하되, 그 배치 영역을 커널이 아닌 사용자 영역에 두어 구현한다. FUSE 기반 가상 파일시스템 환경에서 사용자는 리눅스 고유 유틸리티와 운영체제 서비스를 이용한다.



(그림 2) 사용자 영역의 DB API 기반 가상 파일시스템

3. 리눅스 FUSE 기반 IoT 가상 파일시스템

FUSE[2]는 그림 3과 같이 리눅스 운영체제를 통과하여 사용자 영역의 파일시스템 에뮬레이터를 거쳐 사용자 응용 프로그램의 파일 서비스 요청에 응답하도록 지원한다. FUSE의 실체는 파일시스템을 에뮬레이트하기 위해 구현된 기본 프로시저들의 집합이고(그림 4), 이들 프로시저들은 FUSE 디몬(그림 4의 fuse_main())에 의해 호출되어 사용자가 의도하는 파일시스템 뷰를 유지한다. 그림 5에 POSIX open() 인터페이스에 대응되는 구현 가상 파일



(그림 3) 리눅스 FUSE 개념

시스템의 open() 프로시저를 보였다.

```
static struct fuse_operations vfs_oper = {
    .mkdir   = vfs_mkdir,
    .mknod   = vfs_mknod,
    .getattr = vfs_stat,
    .readdir = vfs_readdir,
    .open    = vfs_open,
    .read    = vfs_read,
    .write   = vfs_write,
    .ioctl   = vfs_ioctl,
    .rmdir   = vfs_rmdir,
    .unlink  = vfs_unlink,
    .flush   = vfs_flush,
    .release = vfs_close,
};
int main(int argc, char *argv[])
{   fuse_main(argc, argv, &vfs_oper, NULL);
}
```

(그림 4) FUSE 구현 프레임워크

4. 구현 가상 파일시스템 기능 검증

그림 6에 가상 파일시스템을 마운트한 후, mkdir 명령어에 의한 디렉터리 생성, mknod 명령어에 의한 노드 생성, ls 명령어에 의한 노드(도메인) 검색, 그리고 cat 유틸리티를 사용하여 디바이스의 실시간 데이터에 접근하는 과정을 보였다. 이 그림에서 'gtwy'는 디바이스 이름이고, 'g'는 디바이스 타입이며, '114.71.70.61:777'은 디바이스의 IP 주소와 포트이다. 이와 같이 리눅스 고유 명령어를 사용하여 파일시스템을 운용할 수 있고, 특히, 표준 입·출력 장치 개념이 적용되는 cat 등 표준 유틸리티를 그대로 사용할 수 있다는 점은 IoT 서비스 개발과정에서 큰 장점이 될 수 있다. 또한 별도의 도메인 관리 툴 없이, 계층 파일시스템을 ls, find 등 친숙한 유틸리티로 검색할 수 있어서 구현이 가볍고 사용이 편리하다.

```
static int iotfs_open(const char *path,
                      struct fuse_file_info *fi)
{
    char   name[L_NAME], type[L_TYPE], iotfd[L_IOTF];
    uint   pinode, inode, uid, gid, mode;
    int    ffd =-1;  time_t ctime;
    if (vfs_namei((char *)path, &inode, &pinode, name,
           &uid, &gid, &mode, &ctime, type, iotfd) < 0)
        return -ENOENT;              // DB query failure
    if (!mode_check(mode, fi->flags) return -EACCESS;

    switch(type[0]) {
    case 'g' :
       ffd = vfs_open_gtwy(name, inode, iotfd); break;
    case 's' :
      ffd = vfs_open_srvr(name, inode, iotfd,
                      fi->flags & O_APPEND); break;
    case 'f' :
      ffd = vfs_open_file(name, inode); break;
    }
    if (ffd < 0) return ffd;  // VFS file descriptor
    fi->fh = ffd; fi->direct_io = 1; fi->nonseekable = 1;
    return 0;
}
```

(그림 5) 구현 가상 파일시스템의 open() 프로시저

```
/> ./dbvfs /tmp/vfs
/> ls -l /tmp/vfs
합계 0
/> mkdir /tmp/vfs/USN
/> mknod /tmp/vfs/USN/gtwy:g:114.71.70.61:7777 p
/> ls -l /tmp/vfs/USN
합계 0
-rw-r--r-- 1 mysql mysql 0 2015-03-13 15:57 gtwy [g]114.71.70.61:7777
/> cat /tmp/vfs/USN/gtwy
T:09,H:67
```

(그림 6) IoT 서비스 플랫폼을 위한 FUSE 기반 가상 파일시스템 운용 모습

5. 결론

이 연구에서는 리눅스 FUSE 메커니즘을 이용하여 IoT 서비스 플랫폼을 위한 가상 파일시스템을 구현하고 기능을 검증하였다. 구현 가상 파일시스템은 id/passwd 등 접근 방법이 다양한 IoT 디바이스 타입별로 대응되는 접근 프로시저를 사용자 영역에서 편리하게 플러그인 할 수 있는 틀을 제공한다. 사용자들은 리눅스 운영체제 인터페이스를 사용하므로 가상 파일시스템을 인식하지 않고 IoT 디바이스를 관리하고 접근할 수 있어서, IoT 서비스 인터페이스 표준화에도 기여할 것으로 기대된다.

참고문헌

[1] Hyung-Bong Lee, Ki-Hyeon Kwon, "Implementation of a DB-Based Virtual File System for Lightweight IoT Clouds", KIPS Transactions on Computer and Communication Systems, Vol. 3, No. 10. Oct. 2014

[2] FUSE, http://fuse.sourceforge.net