시스템 프로그래밍 Device Driver 요약보고서

학번: 201520908
이름: 유성민
학과: 소프트웨어

Device Driver란 특정 하드웨어나 장치를 제어하기 위한 커널의 일부분으로 동작하는 프로그램을 일컫는다. 또는 장치 드라이버라고도 불린다. 왜 커널의 일부 인가하면, 아무나 하드웨어를 조작하면 안되기 때문에 커널의 일부로써 포함되는 것이다. 컴퓨터를 구성하는 다양한 입출력 장치마다 각각 장치 드라이버가 프로그래밍되어 커널에 통합되어 실행된다. 대표적으로는 프린터 드라이버, 마우스 드라이버, 키보드 드라이버 등이 있겠다. 응용프로그램들이 컴퓨터 하드웨어 장치와 상호 작용시키는데 목적이 있다. 만약 Device Driver라는 개념이 존재하지 않았더라면, 응용프로그램을 프로그래밍할 때마다, 하드웨어와 상호작용하는 코드를 집어넣게 되게 이는 큰 문제가 될 것이다. 동작 방식으로는 2가지가 있는데, 첫 번째로는 deviece driver가 커널의 일부분이기는 하나 커널과 통홥되는 것은 처음부터 해당 드라이버 프로그램 코드 소스가 커널 전체 소스에 포함되어 컴파일 되는 경우도 있고, 두 번째로는 별도로 컴파일된 파일의 형태롤 존재하고 부팅 시 또는 필요 시 해당 파일이 로드되어 커널과 통합되기도 한다. 통신방식으로는 device driver는 흔히 bus라고 불리는 하드웨어와 이어진 통신 하위 시스템을 통해 장치와 통신하게 된다. 요청하는 프로그램이 드라이버의 명령어를 호출하면, 드라이버는 장치에 명령어를 전달한다. 장치가 드라이버에게 데이터를 되돌려주면, 드라이버는 원래 요청한 프로그램의 명령어로 데이터를 다시 전달한다. 예를 들어보자. 어떤 응용프로그램에서 키보드를 통해 입력을 받아야 하는 상황을 가정하자. 그렇다면, 응용 프로그램에서는 input에 관한 명령을 수행하는 코드가 실행될 것이다. 그렇다면 키보드를 통해 입력을 받고, device driver는 인터럽트를 날려서 입력이 왔다는 것을 응용 프로그램에게 알리고, device driver의 버퍼에 존재하는 입력받은 데이터를 메모리로 옮겨 프로그램으로 전달하는 것이다. device driver의 특징으로는 하드 웨어에 의존하며, 특정 운영 체제에 따른다 즉, 하드웨어와 운영체제에 종속되는 것이다. 초기의 운영체제 microsoft사의 ms-dos의 경우 하드웨어를 제어하기 위한 어셈블리어를 응용 프로그램에서 직접 사용할 수 있었다. 하지만 이것이 보안성에 굉장히 취약하여 본격적인 운영체제(windows NT계열, UNIX 계열)에서는 커널과 응용프로그램이 분리가 되고, 여러 가지 운영체제가 등장하면서 각기 다른 divice driver가 생겨나게 되었다. 운영체제마다 policy, mechanism이 다르고 설계방식도 다르기 때문에(layerd-approach구조라든가, monotolic 구조, modular 구조 등) 각기 구조가 다르기 때문에 device driver 또한 운영체제에 종속될 수 밖에 없고, device driver는 하드웨어의 칩의 레지스터에 접근하여 제어하기 때문에 각기 다른 회로를 가진 하드웨어에도 종속될 수 밖에 없다. 전자 제품에서 각각의 주변 기기

들을 제어하기 위해 설계된 펌웨어 또한 device driver로 분류된다. 수업에서 다룰 linux 기반 운영체제에서는 /driver 디렉터리 밑에 있는 파일들을 통해 device driver의 소스코드를 볼 수 있다. 설계적인 측면에서 보면 드라이버는 작동하기 위해 하드웨어 기능에 대한 낮은 수준의 접근이 필요하기 때문에, 드라이버는 일반적으로 매우 특권적인 환경에서 작동하며, 문제가 발생하면 시스템 작동 문제를 유발한다. 응용프로그램과는 상당히 대조적인데, user mode에서 실행중인 드라이버도 잘못 프로그래밍 된 경우 시스템 충돌을 일으킬 수 있기 때문에 문제를 진단하는 것을 더 어렵고 위험하게 만든다. device driver의 코드를 작성할때는 유의해야 할 것이다.