포켓리눅스를 이용한 개인 휴대용 정보 단말기 기술

전자부품연구원 손재기* · 이형수
한국외국어대학교 양만석

1. 서 론

최근 리눅스를 PDA 운영체제로 채택한 사례가 늘고 있다. 이러한 이유는 리눅스만의 장점인 오픈 소스라는 점과 두터운 개발자층을 확보하고 있기 때문이다. 리눅스나 다른 무료 오픈 소스 소프트웨어를 PDA에서 돌리는 것은 다른 플랫폼에서 구동하는 것과 같다. 즉 직접 제어가 가능하고 백도어나 많은 프로그램 버그가 없다고 신뢰할 수 있다. 리눅스를 사용하면 30달러까지 하는 윈도우 CE의 부담 없이 장비를 제조할 수 있다. 그래서 많은 PDA 제조업체들이 관심을 가지고 있다. 사실 PDA, 포켓 PC가 등장한 지 얼마 안되었고 리눅스는 기존 운영체제보다도 더 늦게 시장에 발을 들여 놓았기 때문에 아직은 이렇다 할 혁신적인 모습은 찾아보기 힘들다. 또한 우후죽순처럼 생겨 나는 임베디드 리눅스 운영체제들도 여러 가지 표준화 등의 문제를 야기하는 것도 사실이다.

일반적인 개인 휴대용 리눅스 배포본들과는 달리 포켓리눅스는 리눅스 환경 제공보다는 자바의 실행 환경을 제공하는 것을 목적으로 하고 있다. 실제로 프로젝트의 시작이 Kaffe 자바 가상 머신(JVM, Java Virtual Machine)의 제작자에 의해서 진행되었으며 자바의 슬로건이 '개발은 한번만, 실행은 어디에서나'를 기본으로 자바 어플리케이션의 실행 환경을 지원하는데 초점을 맞추고 있다. 본 고에서는 포켓리눅스 기술에 대한 전반적인 내용을 기술한다.

2장에서는 개인 휴대용 단말기와 관련된 임베디드 리눅스 프로젝트 및 임베디드 그래픽 사용자 인터페이스 환경에 대해 기술하고, 3장에서는 포켓리눅스의 전반적인 기술에 대해, 그리고 마지막으로 결론 및 향후 연구과제에 대해 기술한다.

* 정회원

2. 관련 연구

Handhelds 프로젝트

컴팩에서 지원하는 핸드헬드 프로젝트는 주로 일반적인 목적의 배포본들을 개발하는 반면 이와는 달리 특화된 형태로 개발되고 있는 배포본들도 있다.

파밀리아(Familiar) 리눅스

2002년 5월 버전 0.01부터 릴리즈 되기 시작했으며 2001년에 급격한 성장을 이루고 있다. 파밀리아 배포본은 파이썬과 블랙박스 윈도우 매니저를 포함하는 XFree86 등으로 가벼운 운영체제를 만드는데 중점을 두었다. 또한 'ipkg'라는 패키징 시스템을 도입하였는데 이는 RPM, DEB와 같은 데스크탑 리눅스에서 사용되는 패키지 관리자와 같은 역할을 한다.

인티메이트(Intimate) 리눅스

파밀리아 프로젝트의 확장 프로젝트로 파밀리아와 다른 점은 비공식적인 확장을 지원하도록 디자인되었으며 적어도 340MB의 공간을 필요로 한다. 인티메이트의 최종 목적은 '총체적인 환경을 지원하는 배포본'을 제공하기 위한 것이며, KDE Konqueror, MPEG 플레이어와 같은 종합적인 패키지들을 포함한다.

GtkFB

Gtk+는 본래 GIMP라 불리는 영상 처리 프로그램의 그래픽 사용자 인터페이스 환경을 위한 툴킷으로 제작된 GNU 프로젝트였다. 이렇게 만들어진 Gtk+ 그래픽 툴킷은 확장성, 이식성, 안정성이 뛰어나서 기존의 상용 X 윈도우 툴킷을 대체하여 점차 인기를 얻게 되었다. 하지만, 다양한 언어와 위젯들을 지원하는 반면 스레드 기반의 소형 임베디드용으로 쓰기에는 크기가 여전히 큰 편이었으므로, 하부의 X 윈도우 대

신 직접 프레임버퍼에 그래픽 처리를 하도록 구현한 것이 GtkFB이다. GtkFB는 상대적으로 크기가 작은 대신, 다양한 그래픽 장치 가속기들을 지원하지 않으며, 단일 프로세스 모델로 구현된 단점을 가진다[11].

Qt/Embedded

Qt는 trolltech사에서 개발한 C++기반 상용 GUI 툴킷이다. 이것은 사용자에게 고급 위젯들을 많이 제공하고, 국제화 및 지역화가 가능하도록 다중 언어를 지원하고, 리눅스는 물론 윈도우 등의 다양한 플랫폼에 이식 가능한 특징을 가지고 있다. 또한 QPE(Qt Palmtop Envrionment)라 불리는 개인 정보 관리 패키지는 물론, 우수한 개발 도구들을 제공한다. Qt/Embedded는 GtkFB처럼 소형 임베디드용으로 사용하기 위하여 하부의 X윈도우를 쓰지 않고 프레임버퍼에 직접 접근 가능하도록 수정한 것이다. Qt는 라이브러리 소스는 공개하고 있으나, Qt를 이용한 개발 결과를 사용화할 시 자사에서 정한 라이센스 정책을 따라야만 한다[11].

DirectFB

하드웨어 가속을 제공하는 리눅스 프레임버퍼 기술의 또 다른 버전이다. 입력 장치의 추상화, 알파 텍스처, 블렌딩, 그리고 모듈화와 같은 기능을 제공한다.

MiniGUI

윈도우 CE와 유사한 API와 화면 구성을 제공한다. MiniGUI는 MiniGUI-Threads와 MiniGUI-Lite 두 가지 모드를 제공한다. 하나의 프로세스는 서로 다른 스레드에서 윈도우를 생성할 수 있으며 나중에 서로 다른 프로세서에서 각각의 윈도우를 조정할 수 있다. 서버는 Xfree86과 같은 설정을 제공한다.

OpenGUI

마우스, 키보드와 같은 모든 이벤트를 바탕으로 한 2차원 드로잉을 제공하는 시스템이다. 리눅스에서는 프레임버퍼/svgalib를 사용한다.

PicoGUI

C 언어로 작성된 X윈도우와 같은 클라이언트/서버 모델을 바탕으로 한 작은 크기의 그래픽 사용자 인터페이스이다. 클라이언트는 TCP/IP를 통해서 서버와 통신한다. PicoGUI의 가장 큰 특징은 여러 클라이언트가 오버랩되지 않고 서로 다른 윈도우를 생성할 수 있다.

PicoTK

X 서버와 X 라이브러리를 요구하며 X 윈도우 상에서 프레임버퍼로 에뮬레이션 할 수 있다.

Tiny-X

X 윈도우를 적은 크기로 줄인 버전이다. 일번적으로 1MB이하의 크기로 만들 수 있다. Tiny-X는 기존의 X 윈도우에 비해 메모리 부족 현상에 강하다.

Microwindows

Microwindows는 Century Software 사가 임베디드 시스템용으로 개발한 오픈 소스 그래픽 윈도우 시스템이다. 하부의 X윈도우를 대신하여 Nano-X 서버 및 프레임버퍼용 그래픽 프리미티브 라이브러리를 구현하였다[11].

3. 포켓리눅스

포켓리눅스는 자바를 바탕으로 한 어플리케이션 프레임 웍(FrameWork)이다. 자바를 이용하기 때문에 개발자는 쉽게 어플리케이션을 작성하고 배포할 수 있다. 또한 개발자는 XML의 사용으로 어플리케이션을 쉽고 유연성 있게 작성할 수 있다. 수행 성능과 관련된 하드웨어 제어 부분은 자바의 네이티브(native) 코드를 통해 액세스한다.

리눅스의 장점인 이식성을 바탕으로 새로운 장치나 플랫폼에 쉽게 이식할 수 있다. 포켓 리눅스는 PDA나 셀룰러 폰, 페이저, 인터넷 TV와 같은 정보 기기에 효과적으로 이식될 수 있다.

포켓 리눅스는 자바 가상머신 상에서 동작하는 프레임 웍이다. 포켓 리눅스가 동작하기 위해서는 타겟 시스템에 자바 가상머신(JVM)을 포팅하여야 한다. 자바 가상머신의 포팅을 위해 Kaffe OpenVM을 사용하였다. 일반적으로 자바 가상머신은 실행 파일에 비해 실행 속도가 느리다. 또한 사용자 인터페이스 관리자에 의해 그래픽을 출력하기 때문에 사용자 인터페이스 관리자에 의존적이다.

윈도우나 X-윈도우와 같은 사용자 인터페이스 관리자는 임베디드 시스템에 적용하기에는 상당히 덩치가 크다. 자바 가상머신을 임베디드 시스템에 적용하기 위해 자바 가상머신의 그래픽 출력 부분을 수정하여 임베디드 시스템으로 직접 출력할 수 있다.

XML과 자바 기술은 포켓리눅스의 코어부분이다. XML은 화면의 레이아웃을 위해 사용되며 HTML의 테이블 정의와 비슷한 구문을 사용하여 화면을 구성한다. 간단한 어플리케이션의 경우 XML과 관련된 기술로 쉽게 구현할 수 있다. 좀 더 복잡한 어플리케이션의 경우 자바를 사용한다.

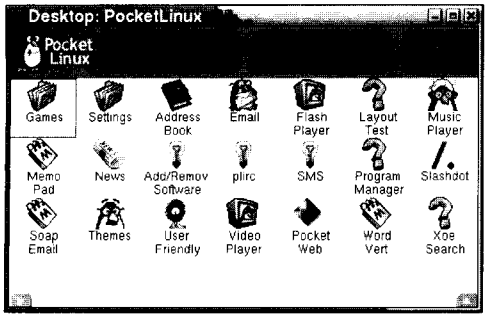


그림 1 포켓리눅스 실행화면(데스크탑)

XOE(eXtensible Operating Environment)

XOE 프레임 웍은 "뚱뚱한 클라이언트" 와 "웹중심"의 응용 프로그램 모델을 하나로 통합하는 모델이다. 정보가전의 넓은 영역을 의미하고, 기존의 데스크탑 시스템을 넘어선 휴대성과 확장성을 추구한다. 소프트웨어적인 측면에서 보면 XOE 프레임 웍은 퍼스널 자바 1.2a 런타임 환경 위에 순수한 자바 라이브러리로 구현되어 있다. 또한 자바 애플릿과 같이 플랫폼 독립성을 보장한다. XOE 프레임 웍의 주요한 목적은 XOE 어플리케이션이 요구하는 내부구조를 제공하기 위함이다.

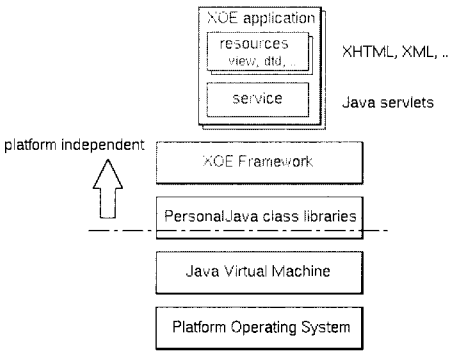
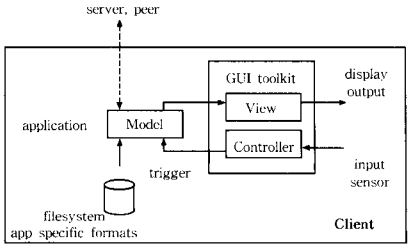


그림 2 포켓리눅스 및 XOE 환경

오늘날에는 두 가지의 주류 어플리케이션 프로그램이 있다. 먼저 첫 번째 것을 살펴 보면

"뚱뚱한 클라이언트"이고 이러한 모델은 이전 인터넷 시대에서 기원하고 주로 데스크 탑 그래픽 유저 인터페이스를 지향했다.

그러나 높은 플랫폼 의존도와 커다란 고정 크기 또는 주로 플랫폼에 링크된 라이브러리와 환경설정 문제와 같은 다양한 문제가 있다.



웹 중심 모델은 공식발표 클라이언트(예: HTML browser)라 가정한다.

어플리케이션 논리에서 가져오는 3가지 주된 방법은 아래와 같다

1. 문서의 문맥에서 실행된 스크립트로 UI 요소에 대한 이벤트 구별하는 것
2. 서버측면 기능성이 활동하는 것 웹서버의 콘텍스트로, 폼 활동을 사용하는 것
3. 문서에 있는 외부 프로그램을 끼워 넣는 것, 이러한 것들은 sandbox 환경에서 동작된다.

XOE 의 가장 중요한 목적은 표준 구조, 형식 그리고 웹 프로그래밍 모델의 구성요소를 사용하는 것이고, 네트웍과 로컬 어플리케이션에 대해서 이것의 사용을 가능하게 하는 것이다.

XOE 의 첫 번째 목적은 로컬 서비스라 불리는 것을 사용해서 주어진 문제를 해결 하는 것이고 두 번째 목적은 XOE 패키지 관리 시스템을 사용해서 주어진 문제를 해결하는 것이다. 자바, XML 그리고 DOM을 사용하는 것은 요구된 휴대성과 상호동작을 확신한다.

웹 프로그래밍 모델에서 클라이언트 측면 각기 다른 로컬 데이터베이스, 그리고 원격 서비스와 함께 연동할 수 있는 어플리케이션 목표화된 프레임 웍 구조이다. 이러한 프레임 웍은 통합된 지원을 최종단 사용자에게 제공한다. 이러한 장점들에 의해서 XOE 의 위치는 잠재력 있는 미래 휴대용 어플리케이션 XML 데이터 프로세싱에 대한 표준화하는 것이다.

로컬 서비스(Local Services)

로컬 서비스들은 XOE 구조의 중추적인 것이다. 게다가 로컬 서비스는 다양한 타입에서 가져올 수 있고 전형적인 어플리케이션 서비스들은 XOE의 논리와 컨트롤을 포함하는 서블릿 인스턴스이다. 이것들은 다음과 같은 것에 의해서 시작하게 된다.

- 컨트롤 이벤트(예: 클릭할 때, 관심대상이 될 때, 키다 눌려졌을 때)
- 폼이 요구될 때 (예: 서블릿들은 FORM 행동을

포함한다.)

· 다른 서비스 들로부터의 요구될 때

이러한 서비스 객체들은 모두 같은 프로세스 안에서 살아있고, 그것들은 서로 다른 것들과 환경의 나머지(예 : 교환된 데이터, 스위치 뷰 문서 등등)와 함께 상호동작 할 수 있다. 지역적으로 이러한 것을 실행하는 관점에서 이것은 서버 측면에서 대응(오직 서버 측면 데이터 또는 (또한) 반환하는 새로운 문서를 수정 하는 것을 가능하게 사용함)하는 것과 비교했을 때 보다 강력한 서비스들을 만들어 낸다.

로컬 서비스는 전체 프레임 웍에 접근할 수 있다. 예를 들면 그들은 문서 스크립트와 같이 똑같이 제한된 것 들은 공유 하지 않는다. 이렇게 함은 대부분의 스크립트 언어에 대한 범위를 넘어가는 랜더링 목적에 대한 지역서비스 하는 것도 가능하게 한다.

개발자들은 오직 한가지 메커니즘과 언어를 다루어야만 한다. 이러한 것들은 또한 다른 자바 컴포넌트 또는 원래 라이브러리와 함께 메커니즘과 언어를 쉽게 인터페이스 하게끔 가능하게 한다. 이러한 서비스들은 표준 서블릿이 될 수 있다. 이러한 보여진 요소와 서비스 사이에서 통신하는데 사용되는 메커니즘은 로컬 동작으로 제한 되지 않는다. 그러나 이것은 또한 전통적 원격 서비스로 사용될 수 있다.

패키지 관리(Package Management)

XOE 는 거대한 단일 어플리케이션에 대해서 사용되는 것을 의미하지 않았다. 응용 프로그램 또는 서비스와 같은 아이디어에 기반해서 설계되었으며 가능한 많은 인프라 스트럭쳐 컴포넌트를 공유해서 작게 유지한다. 모듈간의 의존성은 그 종속성을 따라야만 한다. 모듈에 기반한 구조와 프로세싱은 종단간 사용자에게 투명해야만 한다. 사용자에게 투명성을 제공하기 위해서는 XOE 패키지 관리 서브시스템에 의해 조정된다.

XOE 패키지들은 물리적으로 표준 jar 아카이브, 포함된 XML/XMTML , 의존성에 대한 정보 유지, 충돌, 제공된 기능, 그들에게 일치하는 인스톨 방법처럼 저장된다.

XOE 패키지 관리는 클라이언트 부분(패키지는 탐색과 쿼리를 요구 한다.)과 서버 측면 패키지 저장소를 포함한다, 서버측면 질의시스템은 SOAP 서비스처럼 구현되었다.

일단 모든 요구된 패키지들이 클라이언트로 다운로드되면, 그 패키지 관리자는 설치 요구 순서를 결정하게 되고 특정기능 설치 관리자를 호출해서 그 패키지를 처리 한다. 설치 관리자 클래스들은 환경설정을 하게 되고, 이러한 것들은 그 자체에 의해서 패키지 관리 시스템이 확장할 수 있게 만든다. 설치 관리자 행동들이 자바에서 정의되어져 있고 XOE 프로세스의 안쪽에서 실행되면 이것들은 초기화나 등록하는 자바 객체처럼 복잡한 동작에 대해서 사용할 수 있게 된다. JNI를 사용함으로 인해서 특별한 설치 관리자가 직접적으로 객체화 하는 공유된 원래 라이브러리처럼 플랫폼 구체적인 함수들을 사용하는 것에 대해서 파일시스템에 저장하지 않고 가능하게 한다.

비록 이 프로세스가 유저에 의해서 명백하게 시동된다 해도 이것은 역시 자동으로 오직 유저 상호 동작들이 선택을 다루고 패키지를 설치하는 레벨로 되게 된다.

TVT 프레임버퍼 그래픽 라이브러리(FGL)

대부분의 GUI 툴킷과 윈도우 시스템은 별도의 2 차원 그래픽 라이브러리를 기반으로 한다. 이러한 드라이버들은 특정한 프로세서나 칼라모드, 화면 해상도 등에 초점에 맞춰져 있다. 결과적으로 작은 디스플레이와 적은 메모리, 그리고 전력 소비를 줄여야 하는 임베디드 시스템에 맞지 않다. 대부분의 임베디드 시스템은 간단한 프레임버퍼 디스플레이를 사용하고 특정 디스플레이 프로세서를 가지지 않는다. 반면에 임베디드 시스템은 그래픽 동작을 가속시키기에 적합한 RISC CPU를 활용한다. TVT 프레임버퍼 그래픽 라이브러리는 임베디드 시스템을 고려하여 설계되었다. 사용자 인터페이스 툴킷을 구현하기 위해 요구되는 기본 동작과 특별히 자바 추상 윈도우 툴킷(AWT)를 위해 간소화된 기본 동작들을 포함하고 있다. 프레임버퍼 그래픽 라이브러리는 효과적이고 휴대성이 있는 프레임버퍼 기반의 그래픽에 초점을 맞추고 있다. 입력 장치 관리를 고의적으로 배재하여 입력 프로세싱 라이브러리(IPL)을 따로 제공한다.

FGL은 가공되지 않은 프레임버퍼 디스플레이에서 동작한다. 여기엔 프레임버퍼의 주소를 얻기 위하거나 화면해상도 색상 포맷 등을 얻기 위한 어떠한 운영체제나 하드웨어에 밀접한 저 수준 드라이버 기능이 요구되어지지 않는다. 선, 사각형, 다각형, 원, 타원, 단색 비트맵, 2, 4, 8비트 알파 비트맵, 이미지 등을 지원한다. 폰트는 헤더 파일에 정적으로 링크될 수 있고, 동적으로 메모리 맵핑된 바이너리 파일에서 로딩되거나 또는 트루타입 폰트 파일에서 생성될 수 있다. 유니코드는 폰트 집합에 대한 수단으로 제공되는데, 고 비용적인 로케일들을 요구하는 메모

리의 요구를 상당히 줄인다.

그리기 모드는 소스 복사, 알파 블랜딩 등을 포함한다. FGL은 모든 모양에 대한 간단한 사각형 클리핑을 구현한다. 지원되는 하드웨어 칼라 모델은 4비트 그레이-스케일, 8비트 의사 칼라, 16, 32 비트 트루 칼라, 어플리케이션에 주어지는 논리 칼라 모델은 8888 ARGB이다.

순수 프레임버퍼 접근 레이어 외에, x86, x86 + MMX, ARM과 MIPS에 최적화된 버전이 있다. 하드웨어 가속은 현재 Geode 칩에서 사용 가능하다. 순수 프레임버퍼 접근 레이어와 함께, FGL은 여러 타겟 플랫폼에 포팅될 수 있다. 포팅은 플랫폼과 운영체제 특징적인 프레임버퍼, 칼라 모드 그리고 화면 해상도 질의를 요구할 뿐이다. CPU 특징적인 프레임버퍼 접근 레이어를 제외하고는 시스템은 ANSI C로 쓰여있다. 전체 라이브러리는 60kB(on x86)이하로 컴파일 되고 이는 특정 모듈을 제외하게 되면 더 작게 만들 수 있다.

4. 결 론

현재 많은 분야에서 임베디드 리눅스를 이용한 시스템이 연구 개발 중이다. 본 고에서는 개인 휴대용 단말기에서 효과적으로 사용할 수 있는 포켓리눅스에 대한 전반적인 내용에 대해 기술하였다. 포켓리눅스는 XML과 자바를 이용하여 응용 프로그램을 손쉽게 개발할 수 있으며 확장성 및 이식성이 뛰어나다.

포켓리눅스의 문제점으로 Kaffe 이외의 다른 JVM을 사용하기 위해서는 포켓 리눅스의 엔진(engine) 부분을 수정해야 하며, XML 파서 또한 마찬가지이다. 포켓리눅스가 실제 상용화하기 위해서는 국제화 및 지역화를 위한 부분을 많이 개선해야 한다. 또한 다른 운영체제에 비해 응용 프로그램의 수가 적다. 다수의 응용 프로그램 개발은 포켓리눅스가 성공할 수 있는 조건이다.

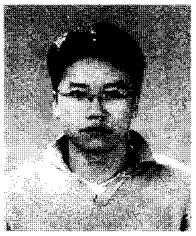
참고문헌

[1] PocketLinux, http://www.pocketlinux.com
[2] Brain Ward, "The Linux Kernel HOWTO", http://www.linux.org
[3] Rerny Card, Eric Durnas, and Framk Mevel, The Linux Kernel book, 2ed., John Wiley & Sons, 1998.
[4] Daniel P. Bovet and Macro Cesati, Understanding the Linux Kernel, O'Reilly & Associates Pub., 2001.
[5] XOE-Extensible Operating Environment, http://www.trasvirtual.com
[6] Kaffe-the Open Source Java Virtual Machine, http://www.transvirtual.com/tech/kaffe.htm
[7] DOM-Document Object Model, http://www.w3.org/XML
[8] SOAP-Simple Object Access Protocol, http://www.w3.org/TR/SOAP
[9] Handhelds, http://www.handhelds.org
[10] 소프트뱅크리서치, "2002 국내 주요 PDA 공급업체 동향분석 및 전망", 2002. 1.
[11] 김성우, 이경우, "임베디드 그래픽 윈도우 시스템 기술", 정보처리학회지 제9권 제1호, pp.76~81, 2002. 1.

손 재 기

2001 경기대학교 전자계산학과 이학석사
2001~현재 전자부품연구원 정보시스템 연구센터 전임연구원
관심분야:네트웍 저장장치, 내장형 시스템, 의료영상처리, 객체지향 프로그래밍
E mail:jgson@keti.re.kr

양 만 석



2002 한국외국어대학교 정보통신공학과 공학학사
2002~현재 한국외국어대학교 컴퓨터 및 정보통신공학과
E mail:msyang@hufs.ac.kr

이 형 수

1989 한양대학교 전자공학과 공학학사
1989~1997 LG전자 미디어통신 연구소
2000 아주대학교 컴퓨터공학과 공학석사
1997~현재 전자부품연구원 정보시스템 연구센터 책임연구원
관심분야:디지털통신 프로토콜, 네트웍 저장장치, 내장형 시스템
E-mail:hslee@keti.re.kr