GTK+기반 Mobile UI Framework

엄예용° 이진석 김호경
삼성전자
{uyysnow°, js708.lee, hkkim.kim}@samsung.com

KISS 33rd Fall Conference

Um Y.Y°, Lee J.S, Kim H.K,
Samsung Electronics

요 약

최근 오픈 소스 소프트웨어를 활용하여 리눅스 기반의 모바일 플랫폼을 구성하려는 추세를 보이고 있다. 특히 최종 사용자(end-user)와의 상호 작용(interaction)을 담당하는 모바일 UI 프레임워크로 GTK+를 많이 사용한다. 본 논문에서는 GTK+의 구조에 대해서 살펴보고, 데스크톱 기반의 GTK+를 모바일 UI 프레임워크에서 사용하기 위하여 고려해야 하는 주요 이슈(issue) 들을 살펴본다.

1. 서론

최근 리눅스(Linux) 운영체제를 기반으로, 오픈 소스 소프트웨어(open source software)를 재사용하여 모바일 기기(mobile device)의 플랫폼(platform)을 구성하는 경우가 늘고 있다[1][2][3]. 오픈 소스 소프트웨어를 이용하면 주요 기능을 구현한 모듈을 확보할 수 있으며, 이들 소프트웨어를 기반으로 만들어진 다양한 어플리케이션(application)을 그대로 사용할 수 있는 이점이 있다. 또한 커뮤니티(community)[4][5][6][7][8]의 활동이 활발한 오픈 소스 소프트웨어를 사용할 경우 지속적인 성능개선을 기대 할 수 있다.

오픈 소스 소프트웨어를 이용한 플랫폼에서 UI 프레임워크(framework)로 GTK+[9]를 많이 사용하는데, GTK+가 사용된 일례로는 Nokia 770 Internet Tablet에 탑재된 Maemo 프로젝트가 있다[10].

앞서 말한 바와 같이, 오픈 소스 소프트웨어는 많은 장점을 가지고 있지만, 이를 모바일 플랫폼에서 잘 활용하기 위해서는 in-house 플랫폼 개발과는 다른 몇 가지 문제점들을 해결해야 한다. 대부분의 오픈 소스 소프트웨어는 소프트웨어의 사용 방법에 관한 문서에 비해 구조를 파악할 수 있는 문서가 부족하다. 또한 데스크톱(desktop)용으로 개발된 소프트웨어는 모바일 환경에 맞게 수정(customize)해야 한다. GTK+ 역시 모바일용 UI 프레임워크에 적용하려면 GTK+의 소스를 직접 분석하여 수정하거나, 기능 추가가 필요한 부분을 파악해야 한다.

본 논문에서는 모바일 기기의 UI 프레임워크 구성에 많이 쓰이는 GTK+를 분석하여 각 모듈간의 관계를 알 수 있는 구조도와, 모바일 기기에 GTK+를 수용했을 때 고려해야 할 이슈를 정리하였다. 운영체제는 GTK+가 가장 유용하게 활용되는 리눅스를 가정하고 GTK+는 2.8 버전을 기준으로 한다.

2. GTK+의 구조 및 기능

GTK+는 GIMP(GNU Image Manipulation Program)프로젝트의 일환으로 GUI(Graphical User Interface)를 작성하기 위해 만들어진 멀티 플랫폼 툴킷(multi-platform toolkit)이다. GNU LGPL 라이센스[11]를 따르고 있으며, 2006년 8월 24일 현재 2.10.1버전이 공개되었다.

GTK+는 X 윈도우 시스템[12], Microsoft Windows, Mac OS X, DirectFB[13], 리눅스 프레임버퍼(framebuffer)에서 실행 될 수 있는데 이중 리눅스의 프레임버퍼를 backend로 사용하는 GTK+를 GtkFB (GTK for Framebuffer)[14]라 한다. GtkFB는 리눅스 커널(kernel)의 프레임버퍼에 직접 접근하므로 메모리가 소요가 적고, 수행 속도가 빠르다. 그러나 싱글 프로세스(single process)기반이므로 멀티 프로세스(multi process) 기반으로 다양한 어플리케이션이 동시에 동작하여야 하는 최근의 모바일 기기에는 부적합 하다. 본 논문에서는 X 윈도우 시스템 기반의 GTK+를 기준으로 살펴보았다.

GTK+를 구성하는 모듈은 GTK, GDK, ATK이다. 이 외에도 Pango, Cairo, Glib, Xlib등 GTK+실행에 필요한 주요 라이브러리가 있다. 그림 1은 이들간의 관계에 대한 구조도이다.

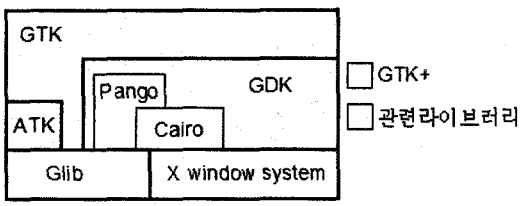

그림1. Overall GTK+ Architecture

각각의 기능과 특징에 대해 간단히 살펴보겠다.

- **GTK**

GUI 구성을 지원하는 툴킷으로 데스크톱 기반의 위젯(widget)과 관련된 150개의 클래스를 지원한다. 위젯은 크게 Window based widget과 No window widget으로 분류된다[15].

- **GDK(GTK+ Drawing Kit)**

저수준(low-level)의 2D 그래픽 함수를 제공하고 X 윈도우 시스템의 상위에 위치한 라이브러리이다. GTK 하위에 다양한 윈도우 시스템(Microsoft Windows, X, MacOS 등)이 올 수 있도록 추상화(abstraction)하는 역할을 한다. 선 그리기, 다각형 그리기, 문자 드로잉과 같은 기본적인 드로잉(primitive drawing) 기능과 중첩 윈도우(overlapping windows)의 제어, 이벤트 생성(event generating)등의 기능을 제공한다. 운영체제에 독립적인 어플리케이션을 개발하기 위해서는 Xlib을 직접 호출하지 않고 GDK를 이용하는 것이 바람직하다.

- **Cairo**

Cairo는 다양한 형태의 출력(output target)을 지원하는 2D 그래픽스 라이브러리이다. 2006년 8월 18일 공개된 1.2.4버전은 X 윈도우 시스템, Win32, 이미지 버퍼(image buffer), PostScript[16], PDF, SVG(Scalable Vector Graphics)[17]형태의 출력을 지원한다.

- **Pango**

Pango는 문자의 레이아웃(layout)과 드로잉(drawing)을 담당하는 라이브러리로 GTK+ 2.0 버전부터 사용되었다. I18N[18] 즉, 다양한 언어를 지원한다. I18N의 Complex 지원 기능의 예로 Arabic 언어의 경우 코드(code)의 입력 순서에 따라 실제 드로잉 해야 할 문자의 형태가 달라지는 특이한 문자가 있으며 Pango는 이를 지원한다.

Pango는 X윈도우 시스템의 폰트 라이브러리인 Xft[19], Cairo, 폰트 랜더링 엔진(font rendering engine)인 FreeType 2[20]의 세가지 랜더러(renderer)를 선택적으로 사용하여 문자를 드로잉 한다.

Cairo는 GTK+ 2.8 버전부터 Pango의 랜더러로 사용되었다.

Cairo를 Pango의 랜더러로 사용하면 Cairo가 지원하는 backend의 형태로 문자를 드로잉 할 수 있다. Cairo가 글립(glyph) 랜더링에 FreeType 2를 사용하기 때문에 Cairo가 지원하는 다양한 형태의 출력이 불필요한 경우 FreeType 2 랜더러를 사용해도 된다.

- **ATK(Accessibility Toolkit)**

ATK는 Accessibility[21]를 제공하는 툴킷으로 장애를 가진 사람들이 GTK+ 어플리케이션과 상호 작용(interaction)할 수 있는 방법을 제공한다. GTK+는 2.0 버전부터 ATK를 구현하여 제공하고 있다. ATK로 구현될 수 있는 상호작용 인터페이스(interface)가 장애인에게만 유용한 것은 아니기 때문에 ATK의 활용시나리오에 대해서도 적극 고려할 필요가 있다.

- **Glib**

GTK+를 이용한 어플리케이션 작성시 유용한 함수와 정의를 제공하는 라이브러리이다. Glib는 GTK+ 프로젝트에서 관리하는 데 malloc()같은 standard libc 함수들을 재정의하고, Double Linked List, String Handling, Error 함수들을 제공한다.

- **Window System**

GTK+는 다양한 윈도우 시스템 기반으로 실행될 수 있다. 리눅스 운영체제에서 사용할 수 있는 윈도우 시스템으로는 X, DirectFB, GtkFB가 있다. 윈도우 시스템의 특징을 파악하여 플랫폼의 성격에 맞는 것을 사용한다.

- **기타**

이 외에도 GTK+를 실행하기 위해서는 JPEG, PNG, TIFF 이미지 로딩(image loading)을 지원하는 libjpeg, libpng, libtiff 라이브러리와 Pango에서 이용하는 FontConfig 라이브러리가 필요하다. FontConfig는 시스템에 설치된 폰트의 위치정보와 폰트 파일들의 목록에 대한 정보를 알려준다.

다음은 GTK+와 주요 라이브러리들 간의 logical view (그림2)이며, 이들 구성요소들간의 관계는 다음과 같다.

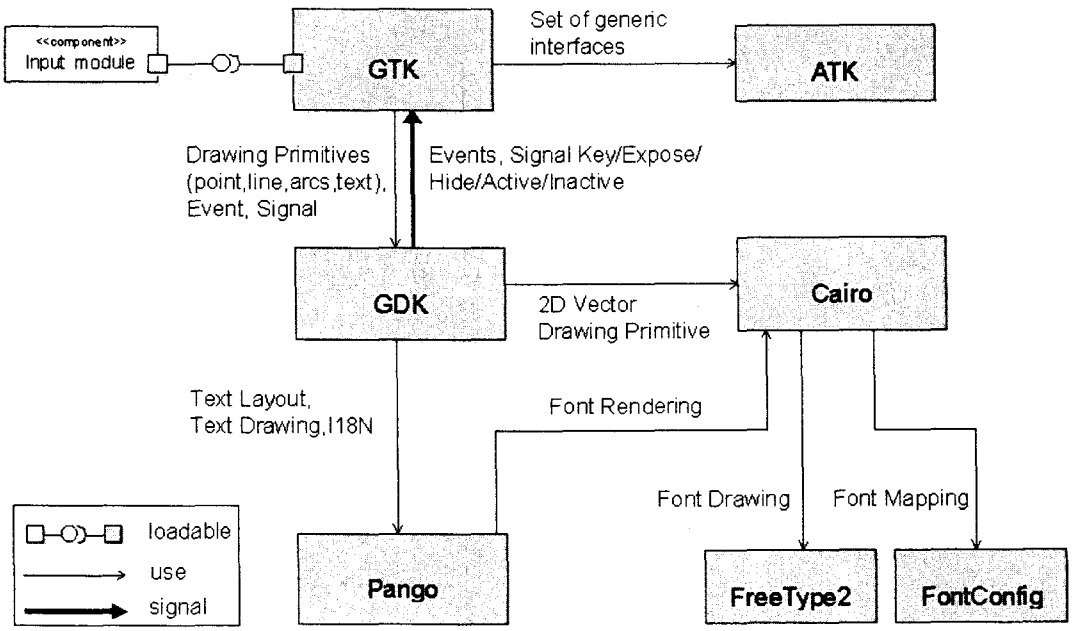

그림2. Logical GTK+ Architecture

- ✓ ATK는 GTK의 각 위젯이 모바일 기기의 최종 사용자와 상호작용할 수 있는 다양한 인터페이스를 제공한다.
- ✓ GTK는 input module로 XIM(X Input Method)을 사용하고 있으나 다른 input module로 교체가 가능하다.
- ✓ GDK는 GTK 위젯을 드로잉 하고, 윈도우 시스템으로부터 전달되는 여러 가지 이벤트를 GTK에 전달한다.
- ✓ 위젯의 문자 드로잉은 Pango 라이브러리를 이용한다. Pango는 Cairo 랜더러를 이용하여 요청된 문자를 드로잉 한다.
- ✓ *Cairo*는 *FontConfig*를 통하여 시스템에 설치된 font list와 위치 정보를 알 수 있고, FreeType 2를 이용해 글립을 랜더링 한다.
- ✓ GDK는 Cairo의 SVG backend를 이용하여 2D 벡터 드로잉을 지원할 수 있다.

이상으로 GTK+를 구성하는 모듈과 관련 라이브러리들의 기능에 대해 살펴보았다. 이 외에도 GTK+ 는 다음의 중요한 기능들을 지원한다. 이 기능들은 GTK+의 여러 모듈에 걸쳐 구현되었다.

- **윈도우 시스템에서 전달되는 이벤트(event)의 처리**

GTK+는 윈도우 시스템으로부터 전달되는 이벤트를 시그널(signal)을 이용하여 처리한다. GTK+가 사용하는 시그널은 리눅스에서 사용하는 시스템 시그널과 다른 것이다.

개발자는 콜백 함수(callback function)를 이벤트가 아닌 시그널에 등록해서 사용해야 한다. 예를 들어 사용자가 특정 위젯에서 마우스를 클릭하면 '마우스 클릭' 이벤트가 윈도우 시스템으로부터 GDK에 전달된다. 사용자가 클릭한 위젯(예, 버튼 위젯)은 '마우스 클릭'에 맞는 시그널(clicked signal)을 발생시키고, 시그널에 등록된 콜백 함수가 실행된다.

- **UI의 look 구성 및 변경**

어플리케이션의 look을 특정 스타일로 표현하는 요소들의 집합을 테마(theme)라 한다. 테마를 적용하면 같은 어플리케이션을 다양한 분위기의 look으로 변경할 수 있다.

테마를 지원하기 위해서는 UI를 구성하는 각종 위젯과 아이콘들이 다양한 색깔과 모양으로 변경될 수 있어야 한다. GTK+에서는 Style, Resource Files, Stock Items, GtkIconTheme등을 이용하여 UI의 look을 구성, 변경할 수 있다.

- **Style**

Style은 위젯의 드로잉에 필요한 graphic context, color, pixmap, colormap, 테마엔진(theme engine)등의 필수 정보를 갖는다. 어플리케이션의 모든 위젯은 각각 Style 정보를 갖고, Style에 지정된 모양과 색깔로 그려지게 된다.

- **Stock Items**

메뉴(menu)나 툴바(toolbar)의 각 아이템에 대응되는 이미지로 각각의 Stock Item들은 ID를 통해 구분이 된다. Stock Items의 경우 데스크톱 성격이 강하기 때문에 모바일 기기의 종류에 따라서 필요 없는 기능일 수도 있다.

- **Resource Files**

Resource file은 Style에 필요한 graphic context , color, pixmap등의 정보와 키 바인딩(*key binding*), 메뉴나 툴바에 쓰일 stock item 등을 기술한다. Resource file에 명시된 정보들을 어플리케이션 실행 시 어플리케이션 전체에 적용할 수 있다.

- **GtkIconTheme**

GtkIconTheme는 각 테마 별로 사용자가 아이콘을 등록하여 어플리케이션에 사용할 수 있는 기능을 제공한다.

- **Drag and Drop**

GTK+는 서로 다른 타입의 프로세스(process)간에 정보를 주고받을 수 있는 selection mechanism을 기반으로 Drag and Drop과 Clipboards기능을 지원한다. 포인터(pointer device)가 지원되지 않는 저가형 핸드폰에는 Drag and Drop 기능이 필요하지 않을 것이다. 윈도우시스템으로 DirectFB를 사용할 경우 GTK+ 2.8버전에서는 아직 Drag and Drop 기능을 지원하지 않는다.

3. GTK+기반 모바일 UI 프레임워크 구성 이슈

데스크톱용으로 개발된 GTK+를 모바일 UI 프레임워크로 활용하고자 할 경우 시스템 환경과 모바일 UI 프레임워크의 기능적/비기능적 요구사항을 만족시켜야 한다.

3.1 시스템 환경

3.1.1 리소스

모바일 기기는 데스크톱보다 낮은 사양의 CPU와 메모리환경을 갖는다. Maemo 플랫폼이 탑재된 Nokia 770 Internet Tablet은 128MB의 플래쉬 메모리와 Texas Instruments OMAP 1710을 사용한다. 이처럼 데스크톱보다 성능이 떨어지는 모바일 기기에 대해 사용자가 만족할 수 있는 성능을 제공하기 위해서는 오픈 소스 소프트웨어를 속도와 크기 측면에서 최적화해야 한다.

UI 프레임워크와 밀접한 관계를 갖는 하드웨어 요소로는 form factor, LCD, keypad등이 있다. 하드웨어의 form factor란 기기의 물리적 케이스(case)의 형태를 의미한다. 핸드폰을 예로 들자면 Candy-bar, Clamshell, Jackknife, Slider, Swivel, Horizontal Instinct 등 다양한 형태의 외관을 가질 수 있다.

LCD는 다양한 해상도와 크기를 가질 수 있으며 form factor에 따라 LCD가 두 개일 수 있다. 핸드폰의 경우 176x220 Quarter CIF, 240x320 Quarter VGA등이 사용된다.

Keypad는 형태가 다양하다. 기기에 따라서 쿼티 키패드(QWERTY keypad)를 제공하기도 한다.

UI 프레임워크 개발자는 모바일 기기의 하드웨어 환경에 맞춰 GTK+의 기능을 수정하거나 새로운 기능을 추가해야 한다. 예를 들어 두 개의 LCD지원, 터치 스크린(touch screen)또는 motion sensor를 통한 최종 사용자의 입력지원, LCD가 회전 가능한 경우 회전 상태에 적합한 가로/세로 기준의 GUI 지원 등의 기능이 추가되어야 한다.

3.2 모바일 UI 프레임워크의 기능적 요구사항

모바일 UI 프레임워크는 데스크톱과 다른 여러 기능적 요구사항이 있다. 요구사항을 만족하기 위해서는 GTK+를 수정하거나 새로운 기능을 추가하여야 한다.

3.2.1 위젯 (Widget)

GTK+의 위젯은 데스크톱기반으로 작성되었기 때문에 불필요한 위젯의 customization이 필요하다. 이때 특정 모바일 기기에서는 데스크톱 성격의 위젯도 필요할 수 있다. 예를 들면 메뉴나 툴바와 같은 데스크톱 성격의 위젯도 스마트폰(smart phone)에서는 필요할 것이다. 따라서 플랫폼이 적용될 모바일 기기의 종류를 결정하여 customize할 위젯을 선택해야 한다.

또한 기기의 특성에 따른 새로운 위젯도 추가해야 한다. 예를 들어 핸드폰의 인디케이터(indicator)는 GTK+에서는 제공하지 않는다. Maemo 프로젝트의 경우 hildon위젯이라 부르는, 플랫폼에 필요한 새로운 위젯들을 추가하였다[22].

3.2.2 어플리케이션 지원

핸드폰은 다양한 개수의 어플리케이션이 작은 화면에서 동시에 수행되어야 한다. 이를 위해 화면에 나타난 어플리케이션에서 다른 어플리케이션으로 전환하는 어플리케이션 스위칭(application switching)과 같은 기능이 제공되어야 한다.

3.2.3 리소스 관리(Resource management)

어플리케이션과 위젯을 구성하는 요소로 문자(text), 색깔(color), 소리(sound), 이미지(bitmap)등이 있다. 모바일 기기에서는 이와 같은 리소스의 집합을 여러 벌 지원할 수 있고, 기기의 최종 사용자가 다운로드 받을 수 있는 서비스를 제공하기도 한다. 메모리 제약을 고려하여 위와 같은 기능을 제공하기 위해 리소스를 관리하는 기능이 필요하다.

3.2.4 테마 관리(Theme management)

GTK+에서는 resource file의 변경을 통해 어플리케이션의 look을 변경할 수 있다. 그러나 모바일 기기에서는 resource file에서 사용하는 리소스보다 더 다양한 리소스를 테마의 요소로 사용할 수도 있으므로 이를 고려한 테마 매니저가 필요하다.

테마 매니저에서는 최종 사용자가 원하는 테마를 다운로드 하거나 어플리케이션에 적용하고, 필요 없는 테마를 삭제할 수 있는 기능을 제공해야 한다.

3.2.5 Predictive input method

GTK+에서는 X가 제공하는 데스크톱용 키보드(keyboard)를 지원하는 IME(Input Method Editor)를 사용한다. 모바일 기기는 키패드(keypad)로 입력이 이루어지고, 같은 문자도 입력 매카니즘이 기기에 따라 다양하다. 따라서 기기에 따라 천지인, T9등의 IME가 지원되어야 한다.

3.2.6 벡터(Vector)/3D 그래픽스(Graphics)지원

모바일 기기의 하드웨어 성능이 향상됨에 따라 벡터, 3D 그래픽스 기반 어플리케이션의 증가하고 있다. UI에 벡터그래픽스 기반의 FlashLite를 적용하여 미려한 화면을 제공하고 3D게임을 제공하는 기기도 있다. 모바일용 UI 프레임워크에서는 GTK+의 2D 그래픽스 외에 벡터/3D 그래픽스의 기능이 지원되어야 한다.

3.2.7 버전 및 backend선택

GTK+를 비롯한 관련 오픈 소스 소프트웨어들은 버전에 따라서 지원하는 기능과, 다른 소프트웨어와의 의존관계(dependency)가 다르다. 동일한 기능에 대해서도 다양한 backend를 지원하기도 한다. 예를 들어 Cairo는 GTK+ 2.8 버전부터 지원이 시작 되었으며, Pango는 Xft, Cairo, FreeType 2의 다양한 랜더러를 제공한다.

개발자는 다양한 backend의 특성과 소프트웨어의 버전 별 지원 기능을 파악해야 한다. 더불어 플랫폼에 필요한 기능을 구체화하면 플랫폼에 적합한 오픈 소스 소프트웨어의 버전과 backend를 선택할 수 있다.

3.3 모바일 플랫폼의 비기능적 요구사항

UI 프레임워크에 한정하지 않은, 모바일 기기용 플랫폼 전체적으로 갖추어야 할 비기능적 요구사항이 존재한다. 그 중 GTK+와 같은 오픈 소스 소프트웨어를 활용할 경우 해결해야 할 주요 이슈는 다음과 같다.

- ✓ **실행성능(Performance):** 낮은 사양의 하드웨어에서 최종 사용자가 만족할 만한 성능을 제공해야 한다. 이에 따른 최적화(optimization)는 필수이다.
- ✓ **유지관리(Maintainability):** 성능 최적화와 모바일에 알맞은 기능 추가를 위해서는 오픈 소스 소프트웨어를 수정해야 한다. 그런데 대부분의 오픈 소스 소프트웨어는 업데이트된 버전을 공개하고 있다. 업데이트된 버전을 사용할 경우 이전에 수정한 코드를 어떻게 반영, 관리할 것인지 고려해야 한다.
- ✓ **사용성(Usability):** 원활한 플랫폼 개발을 위해서는 오픈 소스 소프트웨어의 기능과 새로 추가된 기능에 대한 설명 문서가 필요하다. 또한 오픈 소스 소프트웨어마다 코딩 스타일(coding style)이 다르므로 이를 명시해야 한다.
- ✓ **신뢰성(Reliability):** 오픈 소스 소프트웨어마다 에러 코드(error code)가 각각 다르기 때문에 플랫폼 전체적으로 에러 처리에 대한 통일된 정책을 적용하기 어렵다.
- ✓ **시험 가능성(Testability):** 오픈 소스 소프트웨어 각자 테스트 코드(test code)를 갖고 있다. 이들을 각각 테스트 하지 않고 플랫폼 전체적으로 테스트 하는 쉽고, 통일된 방법이 제공되어야 한다.

4. 결론

GTK+를 모바일 기기용 UI 프레임워크로 활용하고자 하는 개발자를 위해 GTK+의 구조와, UI 프레임워크 구성 이슈에 대해 살펴보았다.

최근 모바일 플랫폼 개발 업계에서는 '오픈 소스 소프트웨어 활용'이라는 새로운 동향이 일고 있다. 이에 동참하기 위해서는 소프트웨어에 대한 기술정보의 공유가 필요하다. 플랫폼 개발자는 오픈 소스 소프트웨어를 활용할 때 생기는 문제점과 장점, 부가적인 개발이 필요한 부분에 대해 고찰해야 한다.

나아가서 오픈 소스 소프트웨어에 대한 분석에만 그치는 것이 아니라 필요한 기능들을 구현하여 이를 적극적으로 소프트웨어의 차기 버전에 반영할 수 있는 영향력을 키워야 한다.

5. 참고

[1] Marguerite Reardon, "Mobile phone companies join forces on Linux", CNET News.com, June 15, 2006

[2] MOAP: https://www.nttdocomo.com/glossary/m/MOAP.html

[3]ALP:http://www.palmsource.com/press/2006/021406_accesslinuxplatform.html

[4] SourceForge:http://www.soruceforge.net,

[5]Freshmeat: http://freshmeat .net

[6]Linux.com: http://www.linux.com

[7]Themes.org: http://www.theme.org

[8]Slashdot:http://www.slashdot.org,

[9] GTK+: http://www.gtk.org

[10] Maemo project site: http://www.maemo.org

[11] GNU's Not Unix: http://www.gnu.org/licenses/

[12] X.Org Foundation: http://www.x.org

[13] DirectFB: http://www.directfb.org/

[14] Alexander Larsson, Gtk+ for the Linux Framebuffer, Mar. 16, 2001

[15] GTK+ Drawing Model:
http://primates.ximian.com/~federico/misc/gtk-drawing-model/index.html

[16] PostScript: http://www.adobe.com/products/postscript

[17] SVG: http://www.w3.org/Graphics/SVG

[18] I18N: http://www.i18nguy.com

[19] Keith Packard, "The Xft Font Library:Architecture and Users Guide", November 8, 2001

[20] FreeType project: http://www.freetype.org/freetype2/index.html

[21] Adobe, What is Accessibility?:

http://www.adobe.com/macromedia/accessibility/gettingstarted/accessibility.html#assistive_technologies

[22]Hildon:http://www.maemo.org/platform/docs/tutorials/Maemo_tutorial.html#Hildon-Widgets