Digital Photography in the GNOME environment

Hubert Figuière <hfiguiere@teaser.fr>

May 14th 2004

1. Introduction

Digital Photography in the last years has been the most dynamic market for photography products. Ever since the first consumer products that appeared in the mid 90s, UN*X users have been attempting to use them with their favorite environment, mostly by performing reverse engineering.

A few open source projects have given photographers ways to access their photos off either digital cameras or scanners from UN*X.

In this paper I'll try to provide background and find the current problems when it comes to GNOME integration. In my talk, I'll try to explain possible solutions to those problem.

2. Acquiring

Inputing photos to the computer is the first step towards working on photos with computers. There are 2 ways to perform this operation: using a digital camera and using a scanner.

2.1. Digital Cameras

Digital cameras are probably the most popular tools in use for photos input. A variety of manufacturers offers a variety of models, and they all provide support for Windows, most for Mac OS, and to date, none provide support for UN*X systems, not even with binary drivers. Since the first models appeared on the market, hackers succeeded in reverse engineering to make them work with their favourite operating system.

2.1.1. Communication protocol

To establish communication between the camera device and the computer, several communication protocols have been developed by the manufacturers, most of them being proprietary and not publicly documented.

Cameras that connect to serial ports were all different, and for each manufacturer, one had to figure out what was the protocol used to download pictures off the camera storage [3]. Sometime, a few

manufacturers bought chips or software to the same OEM vendor, that allowed a few drivers to be more versatile that the others. That was the case for Nikon, Olympus and Epson products that were all using OEM parts from Sierra Imaging. Sometime, we could get the documentation from the manufacturer like Kodak.

Then went USB. USB provided a faster communication pipes to the computer, and USB 1.1 provided a few standardized protocol. The first USB connected cameras were offering most of the time dual USB/Serial connectivity, and the protocol used over USB was usually the same as over serial, making it easy to adapt existing drivers to work with these newer models.

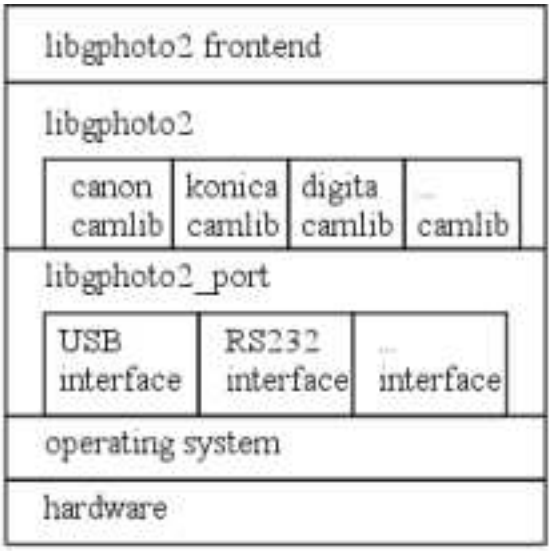
After manufacturers finished transitioning to complete USB connectivity, and after their products matured, standard protocols from USB began to be used more widely: *mass-storage* and later *still image devices* (also known as PTP).

While USB Mass Storage compatible devices are supported natively by Linux 2.4 and other system supporting USB 1.1, simply by mount a filesystem, all the other cameras required some specific drivers, including for PTP, the standard USB Still Image device class.

2.1.2. gphoto

Created in 1999, gphoto has become the de-facto universal swiss army knife tool to access photo from digital camera. The idea was to provide one program for every camera instead of requiring different program for each model available like it was.

When it became gphoto2 in 2000, its internal architecture radically changed to become even more versatile. Everything is now architectured around a library that is modular with different libraries for different cameras, and APIs to write front-ends [1][2].



$Id: architecture.fig,v 1.1 2002/08/17 23:09:55 hun Exp $

camlibs are the libraries that control the camera. This is what you should write or modify to support a new model. They use the *libgphoto2_port* library which abstracts all the I/O through a layer. Current we only have USB and serial support, but there shouldn't be any problem to provide IEEE1394 (aka Firewire) or BlueTooth support later.

libgphoto2 is itself the gphoto2 API. It calls the proper library for the specified device and abstract file retrieval as well as image capture and options setting.

Writing a new front-end is basically calling the APIs from libgphoto2. We currently provide *gphoto2*, a command line front-end, and *gtkam*, a GTK+2 graphical application.

libgphoto2 is currently not meant to provide access to cameras that can simply be mounted like an USB hard disk (USB Mass Storage devices), which represent a majority of the device sold between 2001 and 2003, and still in 2004 [3].

2.2. Scanners

Scanners have been around since mid 80s. They are currently much cheaper than digital cameras, but does not seems that popular for photography use. They are mostly designed to scan flat documents like sheets of paper or printed photos, some, with transparency adapter can scan negatives strips or slides. This is good for scanning archives, but if you want to work quickly from shooting to digital publishing, it is not the best option.

Currently there are very few solutions for scanning under Linux. Almost no manufacturer provide specifications or drivers for Linux, but Epson.

2.2.1. Communication protocols

Unlike for digital cameras, there is no standard protocol either de-facto or not, for scanners. This leads to a bigger development effort. Thanks to the OEM game, only chipsets must be supported as most manufacturers seems to buy them to make their own box. It is not easy to guess what chipset is in a scanner without disassembling it.

Parallel scanners are the harder to support due to lack of traffic analyzer on the parallel port, unlike it is for SCSI, USB and Firewire. Some are just SCSI over parallel ports, some have an USB version that do parallel over USB, allowing to snoop the protocol for reverse engineering.

2.2.2. SANE

SANE stands for *Scanner Access Now Easy*. The project has been around since December 1996 to provide support for scanner devices on UNIX. It offers drivers for a lot of different scanners through different libraries. *libsane* provides the APIs [4] to write front-ends, like *xsane* and *xscanimage*.

Unlike TWAIN, a API really popular in the Windows and Mac OS world, SANE makes the difference between user interface and scanner device drivers. This allow SANE to implement scanning over the network with the *saned* daemon.

2.2.3. Image Scan!

Image Scan! is quite an atypical example. Image Scan! is Espon Kowa own scanner driver for Linux. It is available as Open Source for almost everything but a few bits, like some image processing and drivers for some OEM scanners that Epson sells. Therefore it is limited to Linux on Intel platform. The interesting part is that Image Scan! almost implements SANE APIs making the open source driver usable with plain SANE. And Image Scan! could be modified to use SANE APIs correctly so that the front-end can be used as a SANE front-end.

There is goodwillingness from Epson to continue to develop this program, but there don't seem to be any political wills to push their OEM manufacturers to Open Source the rest of the drivers. Still better than nothing, and the world would be better if all manufacturer where as open as Epson is.

2.2.4. Vuescan

Vuescan is the proprietary and shareware solution to scanning on Linux/x86 (as well as on Windows and Mac OS). It offers support for a lot of scanners models, even some that SANE do not have. It can be your last chance solution. Vuescan also provide good support for scanning negatives as its post-processing routines are much better than what SANE front-ends provide.

3. Managing Pictures

After acquiring all your photos, you probably need to manage them. There are several tools. Amongst them, for GNOME, you have gThumb and F-Spot.

If you want to edit them, there is the famous Gimp. No need to talk about this wonderful photo retouching software.

4. Integration in GNOME

We have SANE and gphoto2 presenting the building blocks for universal scanner and digital camera support, but how do they integrate with GNOME ? Beside Gtkam and Xsane front-ends being written using Gtk+ toolkit, and their availability as Gimp plug-ins, there is not much more.

The first point is getting these to use *libhal* [7]. libhal is not by itself GNOME specific because it target cross-desktop interoperability being part of the *freedesktop.org* initiative. It is one of the key parts of GNOME plug-and-play device support, part of the *Project Utopia* [5], providing hardware abstraction layer.

4.1. Digital cameras

Robert M Love's *gnome-volume-manager* [6] not only provide high-level support for removable storage devices, like playing CD and DVD discs automatically, but it also provide a means to automatically perform and action, like copying photos, when a digital camera is plugged in. Currently it is limited to cameras that behaves like a disk drives, i.e. those that are recognized as *USB Mass Storage* devices [3] and whose content is available as a mounted filesystem. There is some effort to be done on libgphoto2 side to provide support for these mountable camera and use gphoto2 as the "digital photo import utility". That way, we would have automatic photo import when plugging a supported camera ; or any other action the user might prefer.

The other point would be writing a Gnome-VFS plug-in that provide a filesystem like view of the files in the camera using libgphoto2. The

problem is not that simple, as to provide concurrent access to the VFS, we need to serialize all the camera operations by using a daemon that gets exclusive use of the device. This solution is much more elegant even in term of usability, but much more complex to implement.

4.2. Scanning

GNOME does not provide an API to allow acquiring a single image. This is an area that should definitely be worked on.

5. Conclusion

There is still a lot of work to go to have a complete integration for digital camera and scanner high-level support in GNOME. Sure the building bricks are still evolving, but the user side it is not as friendly as one might wish.

6. Bibliography

1. Christophe Barbé — *gphoto2 Foundation for a GNOME photo management application* — http://ufies.org/~christophe/gphoto2/ presented at *Boston Gnome Summit,* July 19, 2002

2. Tim Waugh, Hans Ulrich Niedermann, Michael J. Rensing — *The gPhoto2 Manual* — http://www.gphoto.org/doc/manual/, November 2003.

3. Hubert Figuière — *Digital Camera Support for UNIX, Linux and BSD* — http://www.teaser.fr/~hfiguiere/linux/digicam.html , May 2004

4. The SANE project — *SANE Standard Version 1.03* — http://www.sane-project.org/html/, February 22nd 2003

5. Robert M Love — *Project Utopia* — http://primates.ximian.com/~rml/blog/archives/000395.html, blogged on April 7th 2004

6. Robert M Love — *The Linux Kernel and The Linux Desktop* — http://tech9.net/rml/talks/rml_fosdem_2004.sxi, presented at FOSDEM 2004 in Brussels, February 21st 2004

7. David Zeuthen — *HAL Specification 0.2* — http://freedesktop.org/~david/hal-0.2/spec/hal-spec.html, December 22nd 2003.