OpenPrinting®

cups-filters, CUPS Snap, Driverless Scanning, Printing GUI...

Till Kamppeter – OpenPrinting

17 May 2023

cups-filters: Introduction

- **cups-filters takes up everything from CUPS which Mac OS X does not need** (CUPS 1.6.x)
  - Started end of 2011 by OpenPrinting, **overtaking most of CUPS' filters**
  - Switched filters over from PostScript-centric to **PDF-centric workflow**
  - **cups-browsed** introduced end of 2012, to introduce **browsing of DNS-SD-advertised remote CUPS queues**, as CUPS dropped its own broadcasting/browsing
  - **12** years of **further development** added things like driverless printing support, clustering, support for Printer Applications, IPP standards, PPD-less...

cups-filters Development: Splitting the project

- **cups-filters split into 5 parts now**
  - **libcupsfilters**
    - Filter functions and more for Printer Applications/drivers
  - **libppd**
    - All PPD support of CUPS 2.x and more
  - **cups-filters**
    - Filter/backend executables for CUPS 2.x
  - **braille-printer-app**
    - Printer Application for Braille embossers
  - **cups-browsed**
    - Daemon for printer clusters and legacy printer sharing

cups-filters Development: Splitting the project

- Splitting done to **separate PPD file support**, to easily fade it out, discontinue it in the future
  - **libcupsfilters**
    - Completely free of PPD file support
    - Filter functions take
      - Printer IPP attributes instead of PPDs
      - Job IPP attributes for options
  - **libppd**
    - PPD and PostScript output support for legacy
    - All functionality from CUPS: libcups, ppdc, cups-driverd, cupstestppd
    - For filter functions: Wrapper, PPD → IPP converter

cups-filters Development: Splitting the project

- **cups-filters**
  - CUPS filter/backend executables for CUPS 2.x (legacy)
  - Uses filter functions with PPD support from libppd
  - **`foomatic-rip`** supports PPDs on its own
- **braille-printer-app**
  - Once **converted to Printer Application**, no PPD file support any more
- **cups-browsed**
  - Currently still doing classic queues with PPDs
  - To be **converted to Printer Application**, then free of PPD file support

cups-filters Development: libcupsfilters

- **Filter functions**
  - Converted **all CUPS filters** into **filter functions**
  - Added some **new filter functions**: **`cfFilterPWGToRaster()`**, **`cfFilterUniversal()`**, **`cfFilterExternal()`**
  - Filter functions work **without PPDs**, using **printer and job IPP attributes**)
  - To use filter functions with PPDs, **libppd** provides **data structure extension**, **PPD → IPP attributes converter** and **wrappers**
  - Use **parameters** instead of environment variables
  - All logging into **log function**, no leaks into stderr
  -

cups-filters Development: libcupsfilters

- **Raster data handling**
  - Conversion of image/raster format, color spaces/depth, generate Raster headers, select color space/depth for job
- **IPP Attribute handling**
  - **`get-printer-attributes`**, resolve DNS-SD URIs, select setting from printer/job attributes, select resolutions and page sizes/margins
- **Make/Model/Device ID string handling**
  - Sanitizing, comparing, readability
- **Human-readable strings/translations**
  - Message catalogs from IPP services and from CUPS

cups-filters Development: libcupsfilters

- **Improvements and fixes during last 12 months**
  - All filter and other API functions **totally PPD-independent**
  - Moved all `...toPS()` filter functions and **JCL/PJL** support **to libppd**
  - Lots of fixes and improvements on **page geometry**: Sizes, margins, orientation-requested, ...
  - Support for page sizes **same size but different margins**, borderless, overspray, ...
  - Using **sizes of input pages**
  - Handle also **no printer attributes given**, simply accepting given size, and finally resort to US Letter

cups-filters Development: libcupsfilters

- **Improvements and fixes during last 12 months**
  - On **`get-printer-attributes`** IPP requests try also **`all`** and **`media-col-database`** separately for compatibility
  - **`cfCatalog...()`** API: Support for supplying user language, to allow translations
  - **`cfFilterExternal()`**: Split CUPS/PPD-specific to **`ppdFilterExternalCUPS()`**, support System V interf.
- **Removed functionality**
  - Folded libfontembed into libcupsfilters, removed API
  - Perl and PHP APIs (Nobody used them)
  - Legacy image support (only JPEG, PNG, TIFF stay)
  - PostScript output (moved to libppd)

cups-filters Development: Ghostscript

- **No Artifex presentation this year, but a little bit of Ghostscript news from the CUPS/Apple/PWG Raster output device maintainer (me)**
- **Ghostscript changes** needed for libcupsfilters
  - Direct output of **Apple Raster** (output devices "**`urf`**" and "**`appleraster`**", Ghostscript 10.00.0, commit March 2022)
  - Control **backside orientation**, need of **software copies**, and **CUPS Raster version** via **command line** (was PPD-only before, Ghostscript 10.00.0, commit July 2022)
  - **Do not match custom page sizes against PPD sizes** (Ghostscript 10.01.0, commit October 2022)
  - Feature request posted for **PCLm output in both sRGB and SGray**, got implemented in 10.00.0

cups-filters Development: libppd

- **All PPD file support functionality taken from CUPS 2.x**
  - libcups: All **`ppd...()`** API functions (**`ppd.h`**), made some internal functions public
  - **`ppdc`**: Utilities also some library functions added to API
  - **`cups-driverd`**: Functionality as API functions
  - **`cupstestppd`**: **`ppdTest()`** function, **`testppdfile`** utility
- **Added functionality**
  - PPD support for **filter functions**
    - Convert PPD options/attrib. into printer IPP attributes
    - Wrapper filter functions for special needs
  - Wrapper to turn filter functions into CUPS filters

cups-filters Development: libppd

- **Improvements and fixes**
  - PPD support for **filter functions** (**`ppd-filter.h`**)
    - **`ppdFilterLoadPPD()`**: PPD file → Printer IPP attributes converter (calls also **`ppdLoadAttributes()`**)
    - Wrapper filter functions only for special cases: **`ppdFilterExternalCUPS()`**, **`ppdFilter...ToPDF()`**, **`ppdFilterUniversal()`**
    - Named (“**`libppd`**”) extension for filter function data structure, to hold PPD file name and data, for wrappers
    - **`ppdFilterCUPSWrapper()`**: Wrapper to make PPD-supporting CUPS filter executables from filter functions
  - **`ppdFilterExternalCUPS()`**: Call external, also proprietary, CUPS filters and backends

cups-filters Development: libppd

- **Improvements and fixes during last 12 months**
  - Also **PostScript output is obsolete** and PostScript printers always come with PPD files
    => Move PostScript output filter functions to libppd
    - **`ppdFilterPSToPS()`**, **`ppdFilterPDFToPS()`**, **`ppdFilterRasterToPS()`**, **`ppdFilterImageToPS()`**
  - **`ppdFilterEmitJCL()`**: **JCL/PJL support** in filter functions **moved to libppd**, commands usually supplied by PPD files
  - Support for **page size variants** (**`A4`**, **`A4.Borderless`**, **`A4.Duplex`**, ...), for different margins, sometimes also different print size (overspray).
  - Support for **duplicate sizes** with different names

cups-filters Development: libppd

- **We want to do away with PPDs. Why libppd?**
  - **Legacy printer driver support**
    - No re-writing of driver code for which one has not the hardware for testing
    - Proprietary drivers
    - See also pappl-retrofit
  - **Put together all PPD file support functionality in just one library**
    - Easy fading out of PPD file support
    - Easy discontinuing maintenance
    - Without loss of all the other functionality

cups-filters Development: libppd

- **BUT …**
  - **DO NOT create new PPD files and classic CUPS drivers only because we have this library!**
  - **This library is ONLY for retro-fitting drivers and PPD files which already exist, to avoid re-writing unmaintained, untestable, or proprietary code for devices you cannot buy any more!**
- **If you want to write printer drivers …**

  … create Printer Applications!

cups-filters Development: cups-filters

- **CUPS backend** and **filter** executables for **CUPS 2.x**
- **Filter executables** of general conversion filters **replaced** by small code stub, calling the corresponding **filter function** via **`ppdFilterCUPSWrapper()`**
- Only **`foomatic-rip`** and printer drivers (**`rastertoescpx`**, **`rastertopclx`**, **`commandtoescpx`**, **`commandtopclx`**) are actually implemented in cups-filters
- Backends **`parallel`**, **`serial`**, **`beh`**
- **`driverless`** utility to retro-fit driverless printing support into classic printer setup tools
- **Sample PPD files** for non-IPP PDF printers and PCL-XL printers

cups-filters Development: cups-filters

- **Further improvements and fixes during last 12 months**
  - Streaming mode via **`filter-streaming-mode`** option to run filters optimized or streaming, currently affects **`pdftopdf`**, **`gsto...`**, **`foomatic-rip`** filters.
  - **`imagetops`** implemented via **`ppdFilterImageToPS()`**
  - **`sys5ippprinter`** removed: CUPS does not support System V interface scripts any more, and this first approach of PPD-less printing got never adopted ...
  - **`urftopdf`** removed: CUPS supports URF/Apple Raster by itself
  - **`foomatic-rip`**: Fixed long-standing timing issue when calling Ghostscript to get number of input pages

cups-filters Development: cups-browsed

- **Auto-create local CUPS queues** for network IPP printers and remote, shared CUPS queues
- Create **permanent queues** if CUPS would print with a **temporary queue**, for out-of-date print dialogs
- Create **clusters of printers**, also of different models
  - Automatic for equally-named printers of different servers
  - Manual by config file
- **Legacy CUPS broadcasting/browsing** to connect with CUPS <= 1.5 servers and clients
- Legacy **LDAP** support
- **Highly configurable**

cups-filters Development: cups-browsed

- **Improvements and fixes**
  - Added “`make check`” build tests
  - Fixed **crash bug in multi-threading**, by only creating a thread for Avahi resolving for a newly discovered printer (discovered by build tests)
  - **Fixed resetting of counter** for pausing generation of local queues
  - Fixed **local queues not being removed** when printer disappears
  - Fixed further **crashers**

=> Generally much more reliable now

cups-filters Development: braille-printer-app

- **Not yet released, needs to get converted to Printer Application first**
- Work was started in GSoC 2022 project, GSoC 2023 proposal did not get a contributor slot from Google.
- Need to see whether someone will do it voluntarily.

cups-filters Development: 2.0.0 Release

- **License: Apache 2.0 + (L)GPL2 exception**, same as CUPS
- Cleaned up **naming style to match CUPS:**
  - API functions: “`cfCamelCase()`”, “`ppdCamelCase()`”
  - Library-internal functions: “`_cfCamelCase()`”, “`_ppdCamelCase()`”
  - File-local functions: “`underscore_separated()`”
- Cleaned up all the code to coding style of CUPS
- Bumped **soname** to **2**
- Currently released is **2.0rc1** of all components, used in **Ubuntu 23.04** and **Fedora 38**.
- **2.0.0 final** after testing with **Printer Applications** and **CUPS Snap**

cups-filters Development: Next steps

- **Libcups 3.x support**
  - Planned for **2.1.0**
  - **In 2.0.x fixed API for DNS-SD URI resolution** to easily pass to its support by libcups3.
  - Otherwise **transition will be easy**, we prepared well by the **design of libppd**
  - GSoC contributor Gayatri Kapse (IPP Everywhere 2.x support project) posted already a **pull request for libcupsfilters**

cups-filters Development: Next steps

- **Further plans**
  - Due to our strategy of developing consumers (Printer Applications, ...) and then **developing cups-filters 2.x features as needed by the consumers**, we have reached a **good feature-completeness**.
  - **Future releases** (2.1.0, ...) will happen as new features are needed.
  - **2.1.0** will be for **libcups 3.x support**.

cups-filters Development: Next steps

- **Further feature ideas**
  - **“`cfFilterPDFToPDF()`” PDFio-based** to get rid of C++
  - **cups-browsed** in its own Snap **separate** from the CUPS Snap
  - Turn **cups-browsed** into a **Printer Application**
  - Options for the **`./configure`** script for partial builds: No libqpdf, raster-only printing/scanning, … to **allow Snaps build only the part of cups-filters which they actually need.**
  - **Documentation** of libraries with Mike Sweet’s codedoc utility

The CUPS Snap

- **The “cups” interface, for Snaps of applications which print is complete so far, but**
  - Still uses a content provider (“default-provider”) workaround to auto-install the CUPS Snap
  - Snapd team wants that user gets asked whether they want to install the CUPS Snap on first print attempt
    => Needs further design work on snapd
    => But “cups” can be used though, workaround documented
- Now, with cups-filters 2.0.0 released, we will start a **release/versioning concept** for the CUPS Snap, especially building the Snap based on tags in the CUPS Snap repo. Channels for **upstream** and **Canonical-supported** versions.
- CUPS Snap should be the **printing system of Ubuntu 23.10**, requiring Printer Applications as printer drivers and so we have a **rehearsal for the New Architecture**, before Ubuntu 24.04 LTS with CUPS 3.x
- Available in **Snap Store** “cups”: https://snapcraft.io/cups

The CUPS Snap as a distro's CUPS

- **What is needed:**
  - **DONE: Security concept** on the snapd side completed
  - **DONE: All drivers** available on Debian retro-fitted into **Printer Applications** (only Braille embossers missing)
  - **GUI tools:** GNOME Control Center "Printers" WIP, **CPDB** for PPD-free print dialogs (GTK: DONE; Qt, Mozilla, Chromium, LibreOffice: WIP)
  - **Look-up service for Printer Applications** on OpenPrinting web site planned
    - No follow-up on hardware-look-up feature request for Snap Store
    - Could support also other platforms, like Docker
- **Rehearsal for CUPS 3.x in a distro** (no PPD/driver support)

Driverless Scanning and Scanner Applications

- From the 3 Standards **IPP Scan**, **eSCL**, and **WSD** we will use **eSCL**
  - **eSCL specs published by Mopria**
  - eSCL widely used in **AirScan** devices
  - **IPP Scan not adopted** by scanner industry
- eSCL mainly intended for **multi-function printers**
- **2 SANE drivers** for eSCL: “escl” from Thierry Hucahrd and “airscan” from Alexander Pevzner (also supports WSD), both in most distros
- **eSCL** also works via **IPP-over-USB** (ipp-usb)

Sandboxed Scanner Drivers/ Scanner Applications

- **Current situation: SANE**
  - Scanner driver (**SANE backend**) is shared library
  - Scanning app (**SANE frontend**) links backends dynamically
  - **To add a driver it needs to be dropped in backend dir** => not good for sandboxed packaging
- **New scanning environment: eSCL driverless**
  - Scanner drivers in **Scanner Applications, emulating driverless (eSCL) scanner**
  - Scanning app is **eSCL client**
  - **Legacy:** App uses only sane-airscan SANE backend, SANE drivers enclosed in legacy Scanner Application
  - **Scan support in PAPPL is WIP, GSoC 2022 and GSoC 2023**

Printing GUIs
Print Dialogs

- We need to get **Common Print Dialog Backends** into all dialogs
- **CUPS CPDB backend** then takes care of changes: No use of PPD files, temporary queues, ...
- **Status of dialogs**:
  - **GTK**: Merge request accepted → **DONE**
  - **Qt**: Merge request posted and WIP
  - **Mozilla** (Firefox/Thunderbird): Feature request posted
  - **Chromium**: Design Document created, Feature request to be posted soon
  - **LibreOffice**: Posted on dev mailing list, with reference to a first approach back in 2017

Printing GUIs
Common Print Dialog Backends 2.x

- In the course of **adding CPDB support to the GTK print dialog** GSoC contributor Gaurav Guleria has **added many features to CPDB itself**, leading to the **2.x generation**
- Acquire printer details **asynchronously** (non-blocking)
- **Synchronous** printer data fetching on **backend activation**
- **Backends signal** frontends on **printer updates**
- **Option groups**
- Interfaces for **human-readable/translated** group, option, and setting names
- Retrieve media dimensions from a given “**`media`**” setting
- Support for **margin variants** for the same media size (borderless, ...)
- Support for configurable user and system-wide **default printers**

Printing GUIs
Printer Setup Tool

- **Main Window**
  - List **all IPP print services** as reported by DNS-SD
  - List Printer Applications and their queues or printer/fax in a group
  - No duplicates for IPv4/IPv6, IPPS, interfaces
  - Buttons for web interface, add new queue, show jobs …
- **Add Printer Wizard**
  - List of discovered **non-driverless** USB/network printers
  - Button to see list of **Printer Applications** supporting the printer, installed ones and available in Snap Store (look-up service on OpenPrinting)
  - Buttons to setup printer with selected Printer Application and to install Printer Application from Snap Store
- **Do not remove support for permanent CUPS queues and classic drivers**

Printing GUIs
GNOME Control Center

- Extension of the “**Printers**” module for the New Architecture
- **Main view**
  - **List IPP print services**, each is a print destination for CUPS, without need of actual CUPS queue
  - For MF devices with printer and fax or Printer Applications with various queues, **group the services**.
  - Also list **classic CUPS queues**, for universal compatibility with all CUPS versions.
  - On IPP services button to **open web interface in browser**, no “Set Options”, “Remove printer”, ...
- **Add Printer:**
  - Adding support for finding suitable Printer Applications to **cups-pk-helper**
  - UI to assign and handle both Printer Applications and classic drivers
- **Thanks** to Lakshay Bandlish (GSoC 2020), Divyasheel (2021), Mohit Verma (2022, 2023)
- Also support by the **Canonical Desktop and Design Teams**

Questions / Comments