

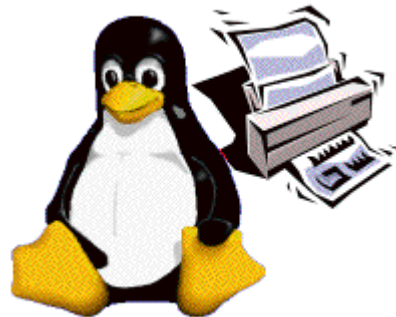


**Free
Standards
Group**

OpenPrinting



*By: Glen W. Petrie
Senior Software Architect
EPSON*





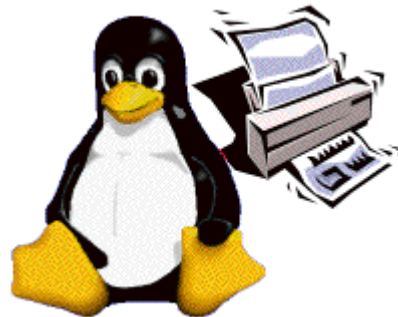
- Introduction
- Working Group Reviews
 - ✗ Architecture
 - ✗ Job Ticket
 - ✗ Application Interface
 - ✗ Driver – Vector/Raster
 - ✗ Status Monitoring
 - ✗ Print Channel Monitor
- Questions





**Free
Standards
Group**

Introduction





Organization



Free Standards Group



OpenPrinting

America/Europe

Japan

Steering Committee

Steering Committee

Architecture

Vector Driver

Job Ticket

Status Monitoring

Application Interface

Print Channel Monitor

Raster Driver





Objectives/Mission

“Standardizing on a
Scalable Print Environment in Linux.”

Mission Statement

The goal of the OpenPrinting is to

develop and promote a set of standards

that will address the needs of

desktop to enterprise-ready printing;

including

management, reliability, security, scalability, printer feature access and network accessibility.

<http://www.openprinting.org/>



❏ OpenPrinting : Input

- ✘ OpenPrinting accepts input from interested parties in IT industry, government, education and the open source community

❏ OpenPrinting : Output

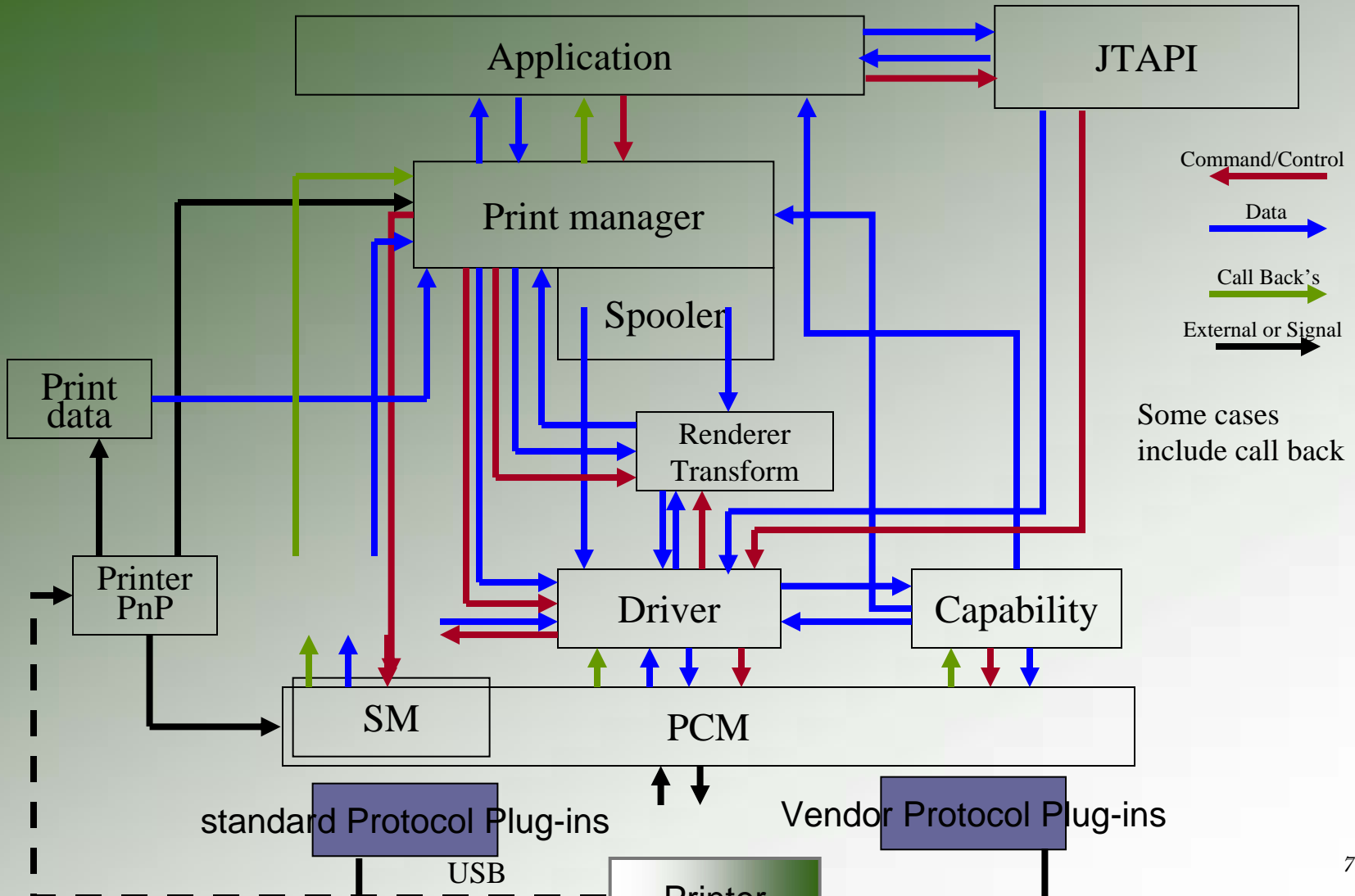
- ✘ Draft specification and implementation released together
- ✘ Public review of work
- ✘ Once implementation has been accepted by *both* upstream authors and at least two Linux distribution vendors, then it is “real”:

A published specification that points to a globally utilized, open source implementation



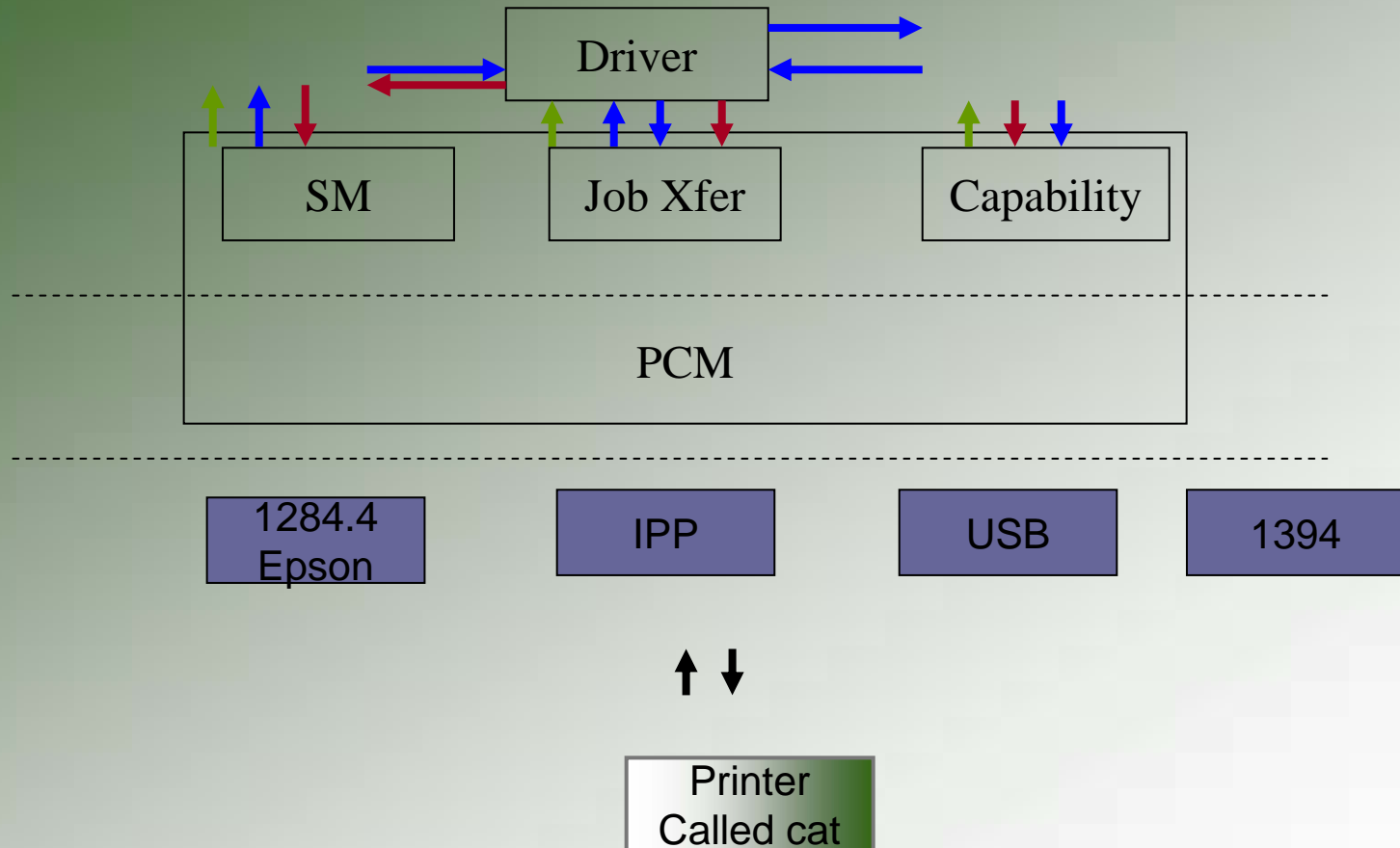
OpenPrinting Reference Model

Preliminary Reference Model





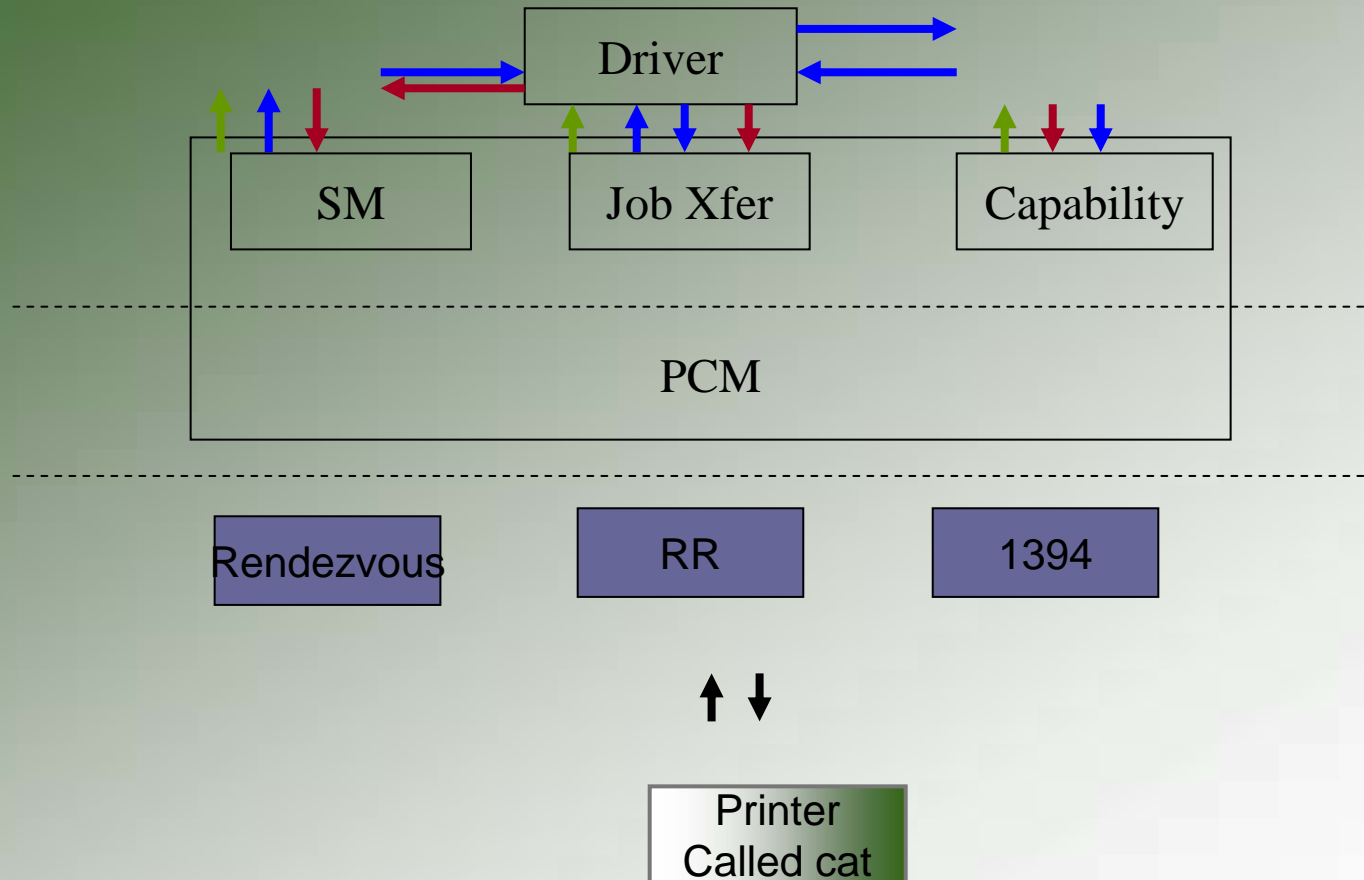
OpenPrinting Reference Model





OpenPrinting Reference Model

Service Discovery model





Working Groups and Objectives

- Steering Committee
 - ✘ Provide overall priorities and top-level coordination.
- Architecture
 - ✘ Develop a modern Print System for Linux.
- Job Ticket API (JTAPI)
 - ✘ To create/consume job tickets; edit job tickets; write/export job tickets.
- Application/Program Interface (PAPI)
 - ✘ Provide applications print services neutral interface.
- Printer Driver (Raster/Vector) (PDAPI)
 - ✘ Provide a neutral interface for printing to any printers
- Status Monitoring API (SMAPI)
 - ✘ Provide a neutral interface for acquiring static and real-time printer status
- Print Channel Monitor API (PCMAPI)
 - ✘ Provide a neutral interface for any output portal/destination.



- Oct 25-26, 2001 – Print Summit Meeting
 - ✗ Began discussion of OpenPrinting and needs.

- June 10, 2003 – FSG Portland Face-to-Face Meeting
 - ✗ Defined Reference Model

- Nov 21, 2004 – FSG San Antonio Face-to-Face Meeting
 - ✗ Working Group Status and Interaction

- Weekly Phone Meetings – all working groups

- Email Communications – all working groups



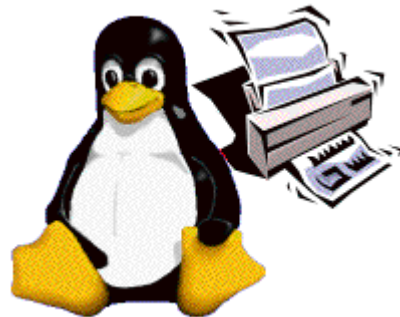
Thanks





**Free
Standards
Group**

Steering Committee





Steering Committee & Objectives

■ Steering Committee

- ✗ Divided into two major teams
 - ◆ US/Europe
 - ◆ Japan

■ Objectives

- ✗ Provide oversight to OpenPrinting Activities
- ✗ Provide direction setting to OpenPrinting Activities
- ✗ Provide coordination between OpenPrinting Working Groups
- ✗ Set priorities for OpenPrinting Activities
- ✗ Release Specification/API from OpenPrinting Working Groups
- ✗ Report to FSG Board on OpenPrinting Activities





Steering Committee Information

- Monthly FSG Steering Committee conference calls
 - ✧ First Monday of each month at 1:00 PM US Eastern for 1-2 hours
- To subscribe to FSG Steering Committee mailing list:
 - ✧ <http://freestandards.org/mailman/listinfo/printing-sc>
- To post a message to FSG Steering Committee mailing list
 - ✧ printing-architecture@freestandards.org
- To view FSG Steering Committee mailing list archives
 - ✧ <http://freestandards.org/mailman/listinfo/printing-sc>
- To find FSG Architecture documents
 - ✧ ftp://ftp.pwg.org/pub/pwg/fsg/steering_committee
- Participants
 - Japan
 - ✧ Takaaki Higuchi (Sun)
 - ✧ Osamu Mihara (FUJI XEROX Printing Systems)
 - ✧ Keisho Shida (Canon)
 - ✧ Yasumasa Toratani (Canon)
 - US / Europe
 - ✧ Mark Hamzy (IBM)
 - ✧ Norm Jacobs (Sun)
 - ✧ Ira McDonald (High North Inc)
 - ✧ Glen Petrie (Epson)





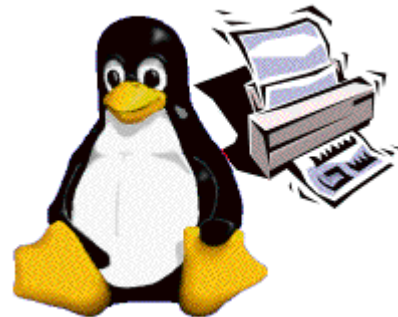
Thanks
from the
Steering
Committee





**Free
Standards
Group**

Architecture





What is OpenPrinting Architecture ?

■ OpenPrinting Architecture is a:

- ✗ Reference Model of the overall solution
- ✗ High-Level Architecture abstracting interfaces between components
- ✗ Detailed Architecture defining interfaces between components

■ Architecture elements contain:

- ✗ Reference Model (RM)
 - ◆ Overall system diagram
 - ◆ Functional decomposition into and description of subsystems.
 - ◆ Identification of data and/or control and/or interface between subsystems and/or external systems.
- ✗ High-Level Architecture (HLA)
 - ◆ A glossary of terminology.
 - ◆ Analysis and documented Use-Cases and requirements.
 - ◆ Identification of applicable/recommended standards for subsystem interfaces.
 - ◆ Identification of recommended infrastructure for integrating products.
- ✗ Detailed Architecture (DA)
 - ◆ Specification of system structure (classes, packages, associations – using UML).
 - ◆ Specification of system behavior (activity and sequence diagrams – using UML).
 - ◆ Identification of the system process structure.
 - ◆ Identification of inter-process communication mechanisms.

■ An OpenPrinting Architecture:

- ✗ Guides development of abstract interfaces.



OpenPrinting Architecture Objectives

- Develop a modern Print System for Linux
- Document realistic Use-Models
- Extract and analyze requirements
- Identify applicable, existing, interface specifications for the requirements
- Apply to multiple print services or be print service neutral.
- Complete and document Reference Model, High-Level Architecture, and Detailed Architecture.



Use-Cases/Use-Models

Example
Details
Exceptions
Diagram
Requirements

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

1. Mobile Printing

Mobile printing by reference with document data transformation.

2. PDA Printing

PDA printing directly with document content.

3. Desktop Personal (Consumer) Printing

Print to low end inkjet printer from an application.

4. Desktop Small-Office/Home-Office Printing

Print to mid-volume laser printer, office inkjet and impact printers.

5. Office Desktop Printing

Document from an application is printed on a printer shared by 3-5 workplaces.

6. Central (Print Room) Office Printing

Mid to large volume printing from document repository with data transformation and post finishing

7. Desktop File (Direct) Printing

Print to print device directly form the file-manager/shell-script without going through an application.

8. Pay-for-Print Printing

Print using a job ticket from a print driver through a print spooler to a mid-range color laser printer.

9. Production Printing

Large volume transactional printing to high-speed production printer.

10. Graphics Art Printing

Large volume journal printing to high-speed printing press.



Mobile Printing Use-Model

1.0 Mobile Printing

Mobile printing by reference with document data transformation.

Example Use Model:

Alice goes shopping for a new digital camera to the Fine Camera store downtown. She takes along a magazine review of the new Bright 3000.

The sales clerk at the Fine Camera store tells Alice that she would probably prefer the Orion 777. Alice uses her mobile hand-held to browse the Web site of Bright and find the URL for the detailed specs of the Bright 3000. She uses the public access printer in the Fine Camera store to print the Bright 3000 specs. Sure enough, the Orion 777 is a better choice.

Details: (1)

Alice turns on her mobile hand-held and hits the 'I' (Internet) button. The mobile hand-held starts a Web browser application, which connects to Alice's wireless Internet service provider (ISP) over a 14.4Kbps cellular modem. Alice types the Bright URL (from the magazine review) into her Web browser and hits the Enter key. The Web browser connects to Bright's home page over the Internet (via the HTTP proxy in the cellular ISP's firewall). Alice searches for the Bright 3000 specs (available in HTML) and copies the URL into her GUI clipboard.

Alice hits the 'P' (Print) button. The mobile hand-held starts a print application, which discovers the store's public access printer over Bluetooth using FSG/OP PAPI (which does Bluetooth device discovery) and then forms an ad-hoc wireless Personal Area Network (PAN) with the Target Printer.

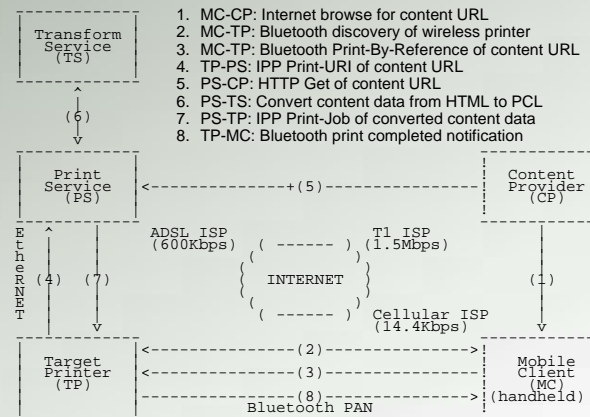


Requirements for this Use Model:

The FSP/OP Architecture MUST support:

- ◆ Administrative configuration of Print Services and their associations with Target Printers using FSG/OP PAPI;
- ◆ Administrative configuration of Target Printers and their associations with Print Services using FSG/OP PAPI;
- ◆ Dynamic discovery of network (IP, Bluetooth, IRDA, etc.) Print Services and Target Printers using FSG/OP PAPI;
- ◆ Multiple print protocol bindings of FSG/OP PAPI (for direct-connect and non-IP network printing);
- ◆ Print-by-reference operations (for example, printing from low-bandwidth mobile devices);
- ◆ Print-by-value operations (for example, printing of converted or local content);
- ◆ Content conversion by FSG/OP Transform Services (for example, reference printing);
- ◆ Job and Printer event notification (for example, job complete, printer intervention required, etc.).

Use Model Diagram:



What's next - schedule



❑ OpenPrinting Requirements Document

- ✗ Aug 2004 - Complete prioritization of identified sub-systems
- ✗ Sept 2004 - Complete Reference Model Alpha Release

**The Architecture has decided to move all discussion to email.
Activities to resume in December**

- ✗ Nov 2004 - Integrate Use Models from FSG-Japan Use-Models
- ✗ Mar 2005 - Complete Detailed Description for Use Models
- ✗ May 2005 - Complete Requirements for Use Models
- ✗ Aug 2005 - Complete Common Requirements & Reference Model 1.0

❑ High-Level Architecture/Reference Model

- ✗ May 2006 - Completed document

How to accelerate the process and schedule?

Funding ! More Participants !





Architecture Working Group Information

- Weekly FSG Architecture conference calls
 - ✦ Thursday at 3:00 PM US Eastern for 1-2 hours
- To subscribe to FSG Architecture mailing list:
 - ✦ <http://freestandards.org/mailman/listinfo/printing-architecture>
- To post a message to FSG Architecture mailing list
 - ✦ printing-architecture@freestandards.org
- To view FSG Architecture mailing list archives
 - ✦ <http://freestandards.org/mailman/listinfo/printing-architecture>
- To find FSG Architecture documents
 - ✦ <ftp://ftp.pwg.org/pub/pwg/fsg/architecture/>
- Participants
 - ✦ Claudia Alimpich (IBM)
 - ✦ Jody Goldberg (Gnome)
 - ✦ Mark Hamzy (IBM) - chair
 - ✦ Tom Hastings (Xerox)
 - ✦ Norm Jacobs (Sun)
 - ✦ Till Kamppeter (MandrakeSoft)
 - ✦ Ira McDonald (High North Inc)
 - ✦ Glen Petrie (Epson)
 - ✦ Pete Zannucci (IBM)





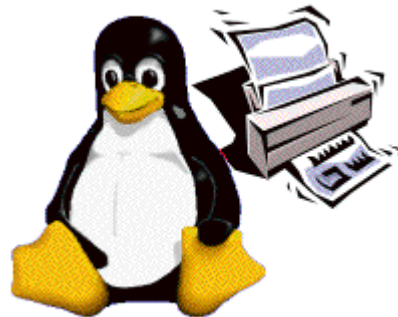
Thanks
from the
Architecture
Team





**Free
Standards
Group**

Job Ticket





JTAPI & Objectives

What is JTAPI ?

✗ JTAPI stands for:

- ◆ Job Ticket Application Programming Interface
- ◆ Pronounced “jay-tappy”, “Job Ticket API”, or “jay tee API”

✗ A job ticket contains:

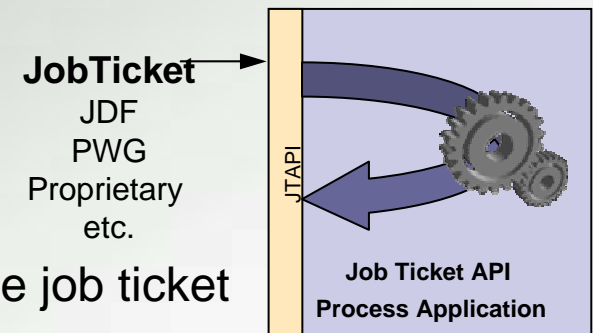
- ◆ **Instructions** describing how to process and/or print a job
- ◆ **Information** about the results of a job as it is processed and/or printed

✗ A JTAPI job ticket is:

- ◆ an electronic replacement of manual hard copy instructions and information

Objectives

- ✗ To create and consume job tickets
- ✗ To be job ticket syntax neutral
- ✗ To isolate the application from the content of the job ticket
- ✗ To be programming language neutral
- ✗ To import and export multiple job ticket formats





Existing Job Ticket Formats

Technical Review

■ CIP4 JDF (Job Definition Format) Job Ticket

- ✘ Defined by CIP4, a world wide standards body with over 150 members
- ✘ Is open, extensible, XML-based job ticket standard
- ✘ JDF Specification versions
 - ◆ 1.0 released April 2001
 - ◆ 1.1 released April 2002
 - ◆ 1.2 released May 2004
 - ◆ 1.3 to be released Mid 2005

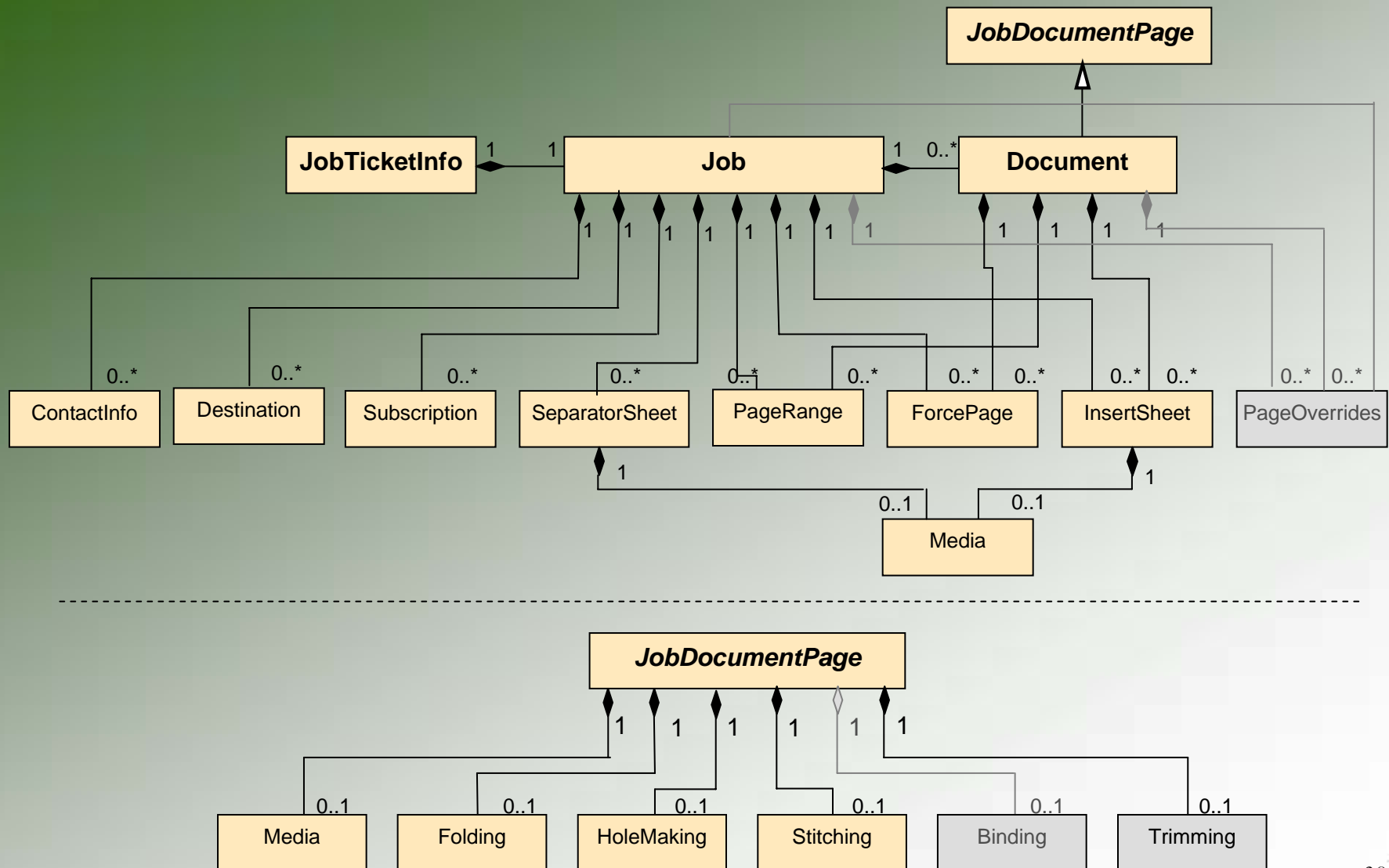
■ PWG Job Ticket

- ✘ Defined by PWG Semantic Model 1.0
- ✘ Is open, extensible, XML-based job ticket standard





JTAPI Object Model





JTAPI Version 1.0 Release

- **Completed C Header Files**
 - ✗ Each object in separate files
 - ✗ Common extensible method for attributes
 - ✗ Data/object model that is object oriented
 - ✗ Defines objects that are familiar to the printing industry
 - ◆ Job, Document, Insert Sheet, Media, Stitching, Hole Making, etc.
 - ✗ Defines relationship between objects
 - ✗ Defines operations to be performed on objects
 - ✗ Defines attributes of objects
 - ✗ Defines well-known enumerated values of all attributes





JTAPI – Version 1.0 Release Details (1)

■ JobTicketInfo

- ✘ **fsgjtNewJobTicketInfo**
 - ◆ Creates a JobTicketInfo object. Used when a new job ticket is being created without a Job object and not from an existing job ticket.
- ✘ **fsgjtNewJobTicketInfoFromURI**
 - ◆ Creates a JobTicketInfo object from the job ticket file at the provided URI. Other JTAPI objects are also created based on the information in the job ticket file.
- ✘ **fsgjtNewJobTicketInfoFromBuffer**
 - ◆ Creates a JobTicketInfo object from the provided buffer that contains a job ticket. Other JTAPI objects are also created based on the information in the job ticket buffer.
- ✘ **fsgjtNewJobTicketInfoFromJob**
 - ◆ Creates a JobTicketInfo object using the provided Job object. Used when a new job ticket is being created. The Job object must be created first.
- ✘ **fsgjtWriteJobTicketToBuffer**
 - ◆ Writes a job ticket to the provided buffer.
- ✘ **fsgjtWriteJobTicketToURI**
 - ◆ Writes a job ticket at the provided URI.

■ Job

- ✘ **fsgjtNewJob**
 - ◆ Creates a Job without providing a Document.
- ✘ **fsgjtNewJobFromDocument**
 - ◆ Creates a Job containing the provided Document

■ Document

- ✘ **fsgjtNewDocument**
 - ◆ Creates a Document object.
- ✘ **fsgjtNewDocumentFromURI**
 - ◆ Creates a Document object using the provided URI that contains the document data.





JTAPI – Version 1.0 Release Details (2)

■ ContactInfo

✘ fsgjtNewContactInfo

- ◆ Creates a ContactInfo object having the specified name.

■ Subscription

✘ fsgjtNewSubscription

- ◆ Creates a Subscription object having the specified notification URI.

✘ fsgjtNewSubscriptionForEvent

- ◆ Creates a Subscription object having the provided notification URI and event.

■ SeparatorSheet

✘ fsgjtNewSeparatorSheet

- ◆ Creates a SeparatorSheet object to be placed at the provided location.

■ ForcePage

✘ fsgjtNewForcePage

- ◆ Creates a ForcePage object having the specified page and sheet side.

■ InsertSheet

✘ fsgjtNewInsertSheet

- ◆ Creates a InsertSheet object.

■ Destination

✘ fsgjtNewDestination

- ◆ Creates a Destination object with a specific URI.



JTAPI –Version 1.0 Release Details (3)

Media

✘ fsgjtNewMedia

- ◆ Creates a Media object using the specified name.

Folding

✘ fsgjtNewFolding

- ◆ Creates a Folding object having the specified folding type.

HoleMaking

✘ fsgjtNewHoleMaking

- ◆ Creates a HoleMaking object having the specified hole count and reference edge.

Stitching

✘ fsgjtNewStitching

- ◆ Creates a Stitching object having the specified stitch type.

PageRange

✘ fsgjtNewRangRange

- ◆ Creates a PageRange object.



JTAPI – Version 1.0 Release Details (4)

Attribute

Generic support for all object/attributes

✘ fsgjtNewAttribute

- ◆ Creates a new Attribute object having the provided attribute name, value type, and value.

✘ fsgjtDestroyAttribute

- ◆ Free the memory used by the Attribute.

✘ fsgjtAddValue

- ◆ Add an additional value to this Attribute.

✘ fsgjtGetName

- ◆ Get the name of the Attribute.

✘ fsgjtGetNextValue

- ◆ Return the Attribute's next value.

✘ fsgjtGetNumValues

- ◆ Get the number of values that the Attribute contains.

✘ fsgjtGetValueType

- ◆ Returns the type of the Attribute's values.

✘ fsgjtReplaceValue

- ◆ Replaces the existing value(s) for this Attribute.

✘ fsgjtResetToFirstValue

- ◆ Reset the iterator to point to the first of the Attribute.





JTAPI – Version 1.0 Release Details (5)

✚ Miscellaneous / Helper

✚ **fsgjtDestory**

- ◆ Free the memory used by the an object.

✚ **fsgjtGet**

- ◆ Get the Attribute having the specific name.

✚ **fsgjtSet**

- ◆ Set the Attribute having the specific name.

✚ **fsgjtSetIntegerAttribute**

- ◆ Convenience function for setting an integer attribute.

✚ **fsgjtSetObjectAttribute**

- ◆ Convenience function for setting an object attribute.

✚ **fsgjtSetObjectAttributeList**

- ◆ Convenience function for setting an object list.

✚ **fsgjtSetStringAttribute**

- ◆ Convenience function for setting a string attribute.

✚ **fsgjtSetStringAttributeList**

- ◆ Convenience function for setting a string list.





JTAPI - Version 1.0 – Enumerations

Enumerations

- ✗ BindTypeEnum
- ✗ BooleanEnum*
- ✗ CollateEnum
- ✗ CompressionEnum
- ✗ ContactInfoRoleEnum
- ✗ FeedOrientationEnum
- ✗ FitPolicyEnum
- ✗ FoldTypeEnum
- ✗ HoldEnum
- ✗ ImageAlignmentXEnum
- ✗ ImageAlignmentYEnum
- ✗ InputTrayNameEnum*
- ✗ InsertSheetContentEnum
- ✗ JobTicketTypeVersionEnum*
- ✗ JogOffsetEnum
- ✗ LengthUnitEnum*
- ✗ MandatoryAttributesEnum
- ✗ MediaCoatingEnum
- ✗ MediaColorEnum
- ✗ MediaPrePrintedEnum
- ✗ MediaTypeEnum*
- ✗ OutputBinEnum
- ✗ PageDeliveryEnum
- ✗ PositionEnum
- ✗ PresentEnum
- ✗ PresentationDirectionEnum
- ✗ PrintContentOptimizeEnum
- ✗ PrintQualityEnum*
- ✗ ReferenceEdgeEnum
- ✗ RotationEnum*
- ✗ SeparatorSheetEnum
- ✗ SheetSideEnum
- ✗ SidesEnum*
- ✗ StitchingTypeEnum*
- ✗ SubscriptionEventEnum
- ✗ TrimmingTypeEnum
- ✗ ValueTypeEnum*



Accomplishments 2002

Introduction

- Feb 2002
 - ✗ Began job ticket discussions in FSG Open Print

- June 2002
 - ✗ Initial JTAPI proposal
 - ✗ Chartered FSG JT working group

- Nov 2002
 - ✗ Created IPP to JDF mapping table
 - ◆ In cooperation with PODi and CIP4 Digital Printing working groups

- Dec 2002
 - ✗ Prioritized features/functions of JTAPI
 - ✗ IBM ships first generation C JTAPI product
 - ◆ In a job submission GUI and printer control unit product
 - Based on initial JTAPI specification
 - Based on early version of JDF ICS Specification for Digital Printing



Accomplishments 2003

Introduction

- **May 2003**
 - ✗ Completed detailed JTAPI UML diagrams
 - ◆ 19 objects and 33 enumerations
 - ◆ 20+ draft versions

- **June 2003**
 - ✗ Defined subset of JTAPI 1.0 content for Alpha version

- **Nov 2003**
 - ✗ Completed version 0.82 JTAPI Specification
 - ✗ Started “C” language header files
 - ◆ Initial contribution from IBM

- **Dec 2003**
 - ✗ IBM ships second generation C JTAPI product
 - ✗ IBM ships first generation Java JTAPI product



Accomplishments 2004

■ JTAPI Software Development White Paper

- ✘ May 2004 - Draft white paper completed
- ✘ May 2004 - Completed and distributed to FSG/Japan

■ JTAPI Version 1.0 Specification

- ✘ Oct 2004 - Updated JTAPI global UML diagrams
- ✘ Oct 2004 - Updated JTAPI Version 1.0 compliant C header files
- ✘ Oct 2004 - Updated Version 1.0 Specification
- ✘ Nov 2004 - Review Specification

■ JTAPI Version C Header Files

- ✘ Nov 2004 - Review headers
- ✘ Dec 2004 - Finalize requirement / definition document

■ JTAPI Version 1.0 Specification/Header Release

- ✘ Jan 2005 - Release Specification and headers



JT Working Group Information

- Weekly FSG Job Ticket conference calls
 - ✦ Tuesdays at 3:00 PM US Eastern for 1-2 hours
- To subscribe to FSG Job Ticket mailing list:
 - ✦ <http://freestandards.org/mailman/listinfo/printing-jobticket>
- To post a message to FSG Job Ticket mailing list
 - ✦ printing-jobticket@freestandards.org
- To view FSG Job Ticket mailing list archives
 - ✦ <http://freestandards.org/mailman/listinfo/printing-jobticket>
- To find FSG Job Ticket documents
 - ✦ <ftp://ftp.pwg.org/pub/pwg/fsg/jobticket/>
- Participants
 - ✦ Claudia Alimpich (IBM) – chair
 - ✦ Jody Goldberg (Gnome)
 - ✦ Tom Hastings (Xerox)
 - ✦ Till Kampeter (Mandrakesoft)
 - ✦ Ira McDonald (High North)
 - ✦ Glen Petrie (Epson)



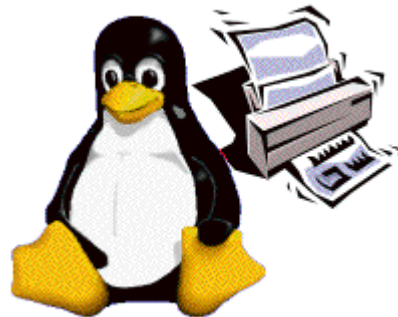
Thanks
from the
JTAPI Team





**Free
Standards
Group**

Application Interface





PAPI & Objectives

■ PAPI stands for:

- ✗ Printing Application Programming Interface
- ✗ Pronounced “pappy” or “P.A.P.I.”

■ PAPI contains:

- ✗ **Data Structures** for interacting with basic print service objects.
- ✗ **Functions** to perform various operations against a print service or its objects.

■ PAPI is an API for applications to use to perform print service interaction.

■ Objectives

- ✗ Provide applications a means of interacting with print services without being tied to a particular service or protocol
- ✗ Allow a rich, extensible set of information to flow between application and print service
- ✗ Support a rich enough set of operations to be useful to most applications with printing needs



PAPI Overview

- ❏ Provides abstraction of server, queues (printers and classes), and jobs.
- ❏ High-level API focuses on providing a common interface to multiple printing systems and protocols.
- ❏ Uses extensible attribute sets based on IPP for representing queue and job information.
 - ✗ Multi-valued and typed
- ❏ Currently supports
 - ✗ Server connection (and authentication were applicable)
 - ✗ Basic queue operations:
 - ◆ enumerate, query, enable, disable, modify, enumerate/purge jobs
 - ✗ Basic job operations:
 - ◆ submit, stream, query, cancel, hold, release, modify



PAPI Details

✚ Data types introduced by the PAPI:

- ✘ **papi_status_t** status/error code enumeration
- ✘ **papi_service_t** server/service connection
- ✘ **papi_printer_t** printer object/handle/context
- ✘ **papi_job_t** job object/handle/context
- ✘ **papi_attribute_t** object attributes
- ✘ **papi_stream_t** stream object/handle/context
- ✘ **papi_job_ticket_t** job ticket
- ✘ **papi_filter_t** filter for limiting printer lists

✚ Attribute API :

- ✘ **papiAttributeListAdd / papiAttributeListAdd***
 - ◆ Adds a single attribute (and value) to an attribute list
 - ◆ Supported types include- String, integer, boolean, range, resolution, datetime, collection, metadata
- ✘ **papiAttributeListGet***
 - ◆ Retrieve the value associates with an attribute
- ✘ **papiAttributeListFind**
 - ◆ Search for an attribute in a list
- ✘ **papiAttributeListGetNext**
 - ◆ Walk through an attribute list
- ✘ **papiAttributeListFromString**
 - ◆ create/append an attribute list using a text-based format
- ✘ **PapiAttributeListToString**
 - ◆ Convert an attribute list to a text base string form
- ✘ **papiAttributeListFree**
 - ◆ Deallocate resources associated with an attribute list

✚ Service API

- ✘ **papiServiceCreate**
 - ◆ create a new service object for submitting jobs, listing printers, etc
- ✘ **papiServiceDestroy**
 - ◆ destroys a service object and all associated resources
- ✘ **papiGet*() / papiSet*()**
 - ◆ Get or set the corresponding service object attributes (UserName, Password, Encryption, AuthCB, AppData)

✚ Miscellaneous API :

- ✘ **papiStatusString**
 - ◆ Return a textual representation of a papi_status_enumeration
- ✘ **papiLibrarySupportedCalls**
 - ◆ Enumerate all papiCalls that do not simply return PAPI_OPERATION_NOT_SUPPORTED
- ✘ **papiLibrarySupportedCall**
 - ◆ Determine if a specific function is supported by the implementation





PAPI Details

Job API:

- ✗ **papiJobSubmit / papiJobSubmitByReference**
 - ◆ Submit a print job either immediately copying or postponing copy of the job data
- ✗ **papiJobValidate**
 - ◆ Verify that the document format and job attributes are valid for and supported by the designated print queue
- ✗ **papiJobStreamOpen / papiJobStreamWrite / papiJobStreamClose**
 - ◆ Open/write/close a stream for printing data on-the-fly
- ✗ **papiJobQuery**
 - ◆ Request job information from print service
- ✗ **papiJobModify**
 - ◆ Modifies the job attributes
- ✗ **papiJobCancel**
 - ◆ Cancels a print job
- ✗ **papiJobHold / papiJobRelease / papiJobRestart**
 - ◆ Hold, release, or restart a print job
 - ◆ Retrieve the corresponding attribute list from a job object for further lookup or enumeration using the Attribute interface
- ✗ **PapiJobPromote (new)**
 - ◆ Promote a print job
- ✗ **papiJobGetAttributeList**
 - ◆ Retrieve the corresponding attribute list from a job object for further lookup or enumeration using the Attribute interface
- ✗ **papiJobGet**
 - ◆ Retrieve the corresponding PrinterName, Id or JobTicket from a job object
- ✗ **papiJobListFree / papiJobFree**
 - ◆ Free a job object of list of job objects and their associated resources

Printer API :

- ✗ **papiPrintersList**
 - ◆ Retrieves a list of printer/class queues
- ✗ **papiPrinterQuery**
 - ◆ Queries for detailed information about a printer object
- ✗ **PapiPrinterAdd / papiPrinterRemove (new)**
 - ◆ Creates/destroys printer object
- ✗ **papiPrinterModify**
 - ◆ Sets printer object attributes
- ✗ **papiPrinterPause / papiPrinterResume**
 - ◆ Stop and start job processing on a printer object
- ✗ **papiPrinterEnable / papiPrinterDisable (new)**
 - ◆ Stop and start job queueing on a printer object
- ✗ **papiPrinterPurgeJobs**
 - ◆ Cancels all jobs on a printer
- ✗ **papiPrinterListJobs**
 - ◆ Lists jobs on a printer
- ✗ **papiPrinterGetAttributeList**
 - ◆ Retrieve an attribute list from printer object
- ✗ **papiPrinterListFree / papiPrinterFree**
 - ◆ Deallocate resources associated with a printer object or list of printer objects





PAPI Status & Plans

■ Status

- ✗ Published v0.91 of specification
- ✗ Released PAPI based code on SourceForge
 - ◆ PAPI dynamic library
 - ◆ PAPI over CUPS
 - ◆ BSD/SYSV print commands implemented on top of the PAPI
- ✗ Released PAPI over RFC-1179 support

■ What's next

- ✗ Expand functionality in PAPI
 - ◆ Administrative operations
 - ◆ Document object
- ✗ Implement and Release PAPI over IPP support
- ✗ Release IPP server implemented over PAPI
- ✗ Integrate support for various Open Source projects:
 - ◆ GNOME, KDE, Open Office, Mozilla, Samba, etc.



PAPI Working Group Information

- To subscribe to FSG PAPI mailing list:
 - ✗ <http://freestandards.org/mailman/listinfo/printing-spool>
- To post a message to FSG PAPI mailing list
 - ✗ printing-spool@freestandards.org
- To view FSG PAPI mailing list archives
 - ✗ <http://freestandards.org/mailman/listinfo/printing-spool>
- To find FSG PAPI documents
 - ✗ <http://sourceforge.net/projects/openprinting>
- Participants
 - ✗ Norm Jacobs (Sun) – chair
 - ✗ Alan Hlava (IBM)
 - ✗ Mike Sweet (Easy Software)
 - ✗ Ira McDonald (High North Inc)
 - ✗ Glen Petrie (Epson)



Thanks
from the
PAPI Team





**Free
Standards
Group**

Vector Printer Driver

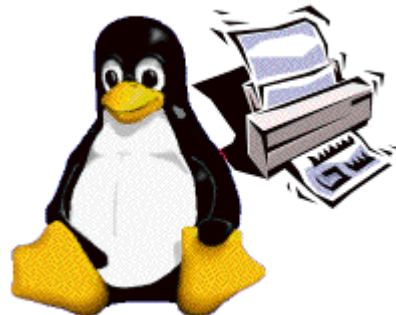
Osamu MIHARA

<mihara.osamu@fxpsc.co.jp>

OpenPrinting WG Japan/Asia

Fuji Xerox Printing Systems Co. Ltd.

15-17 November 2004





What is a Vector Printer Driver?

- Called by render engine, such as Ghostscript or X print server, to convert spool data to PDL.
- Generates PDL using higher level graphics commands, instead of rasterized bitmap image.



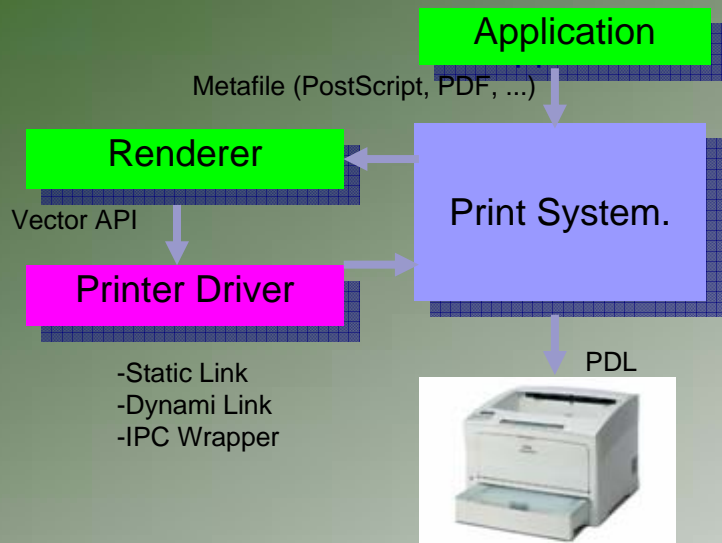
Objectives

- Performance Optimization
 - ✗ Achieve full speed printing on fast laser printers
 - ✗ Utilizes graphical acceleration feature supported by printer controllers
- Data Size Optimization
 - ✗ Reduces size of print data using high level graphics commands.
 - ✗ Contributes to reduce network bandwidth and increase through-put
- Print Quality Optimization
 - ✗ Utilizes printer's graphics quality enhancement technology by sending vector graphics command
 - ✗ Color Optimization
 - ◆ Driver can recognize the kind of graphics primitives and switch color scheme – natural color for bitmaps and vivid colors for graphics and text.
- Independent Design from Rendering Engine
 - ✗ Single driver architecture can be adopted to various printing environment
- Free from Free Software License Woe
 - ✗ Vendor drivers can be provided without making source code open

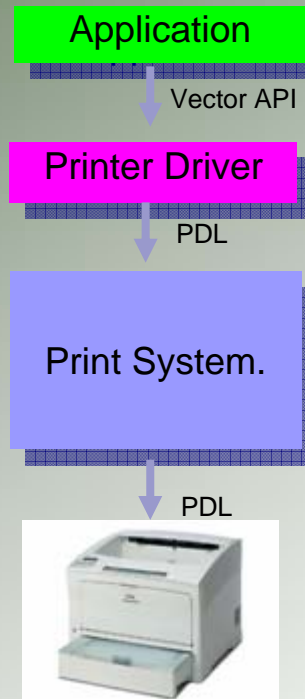


Various Configuration for Vector Printer Driver

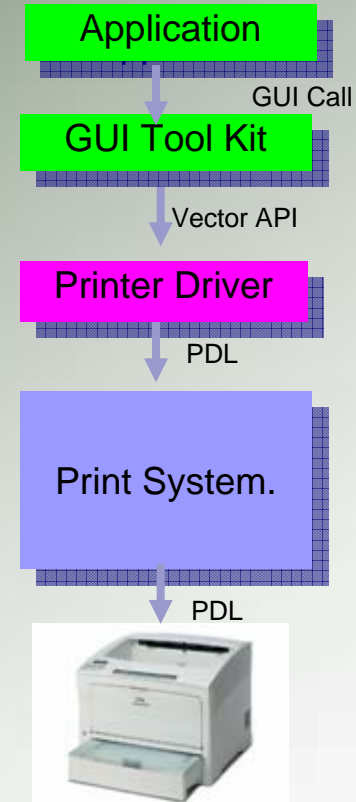
(1)



(2)



(3)



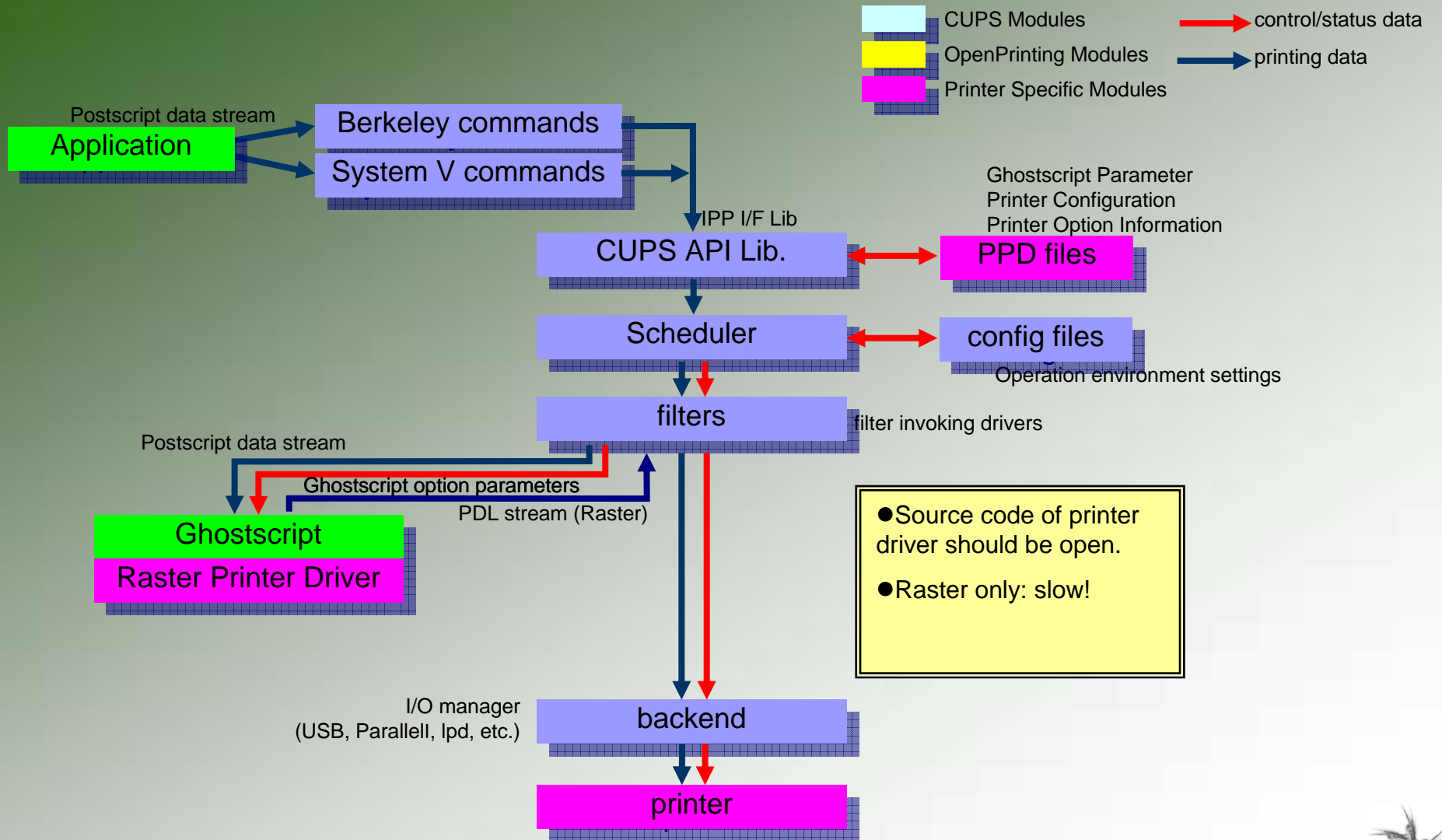


Current Status

- Current Specification Version - Version 0.2
 - ✗ <ftp://ftp.pwg.org/pub/pwg/fsg/vector/>
- opfc implementation based on v0.2
 - ✗ Project funded by IPA
 - ✗ HP PCL5, Epson ESC/Page, Canon LIPS IV
 - ✗ Support on Ghostscript and Xprint
 - ✗ Easily ported to BSD platform
 - ✗ <https://sourceforge.jp/projects/opfc/>
- Current Activity
 - ✗ Device Font & Font downloading support
- To-Dos
 - ✗ Generalize Job Property and Device Capability Parameter definition
 - ✗ Small Memory Device Support (binary encoding of parameters)
 - ✗ Multiple Job for Single Driver
 - ✗ Dynamic APIEntry
 - ✗ Architecture independent data types
 - ✗ (Color Issue)
 - ✗ (formalize IPC Protocol)

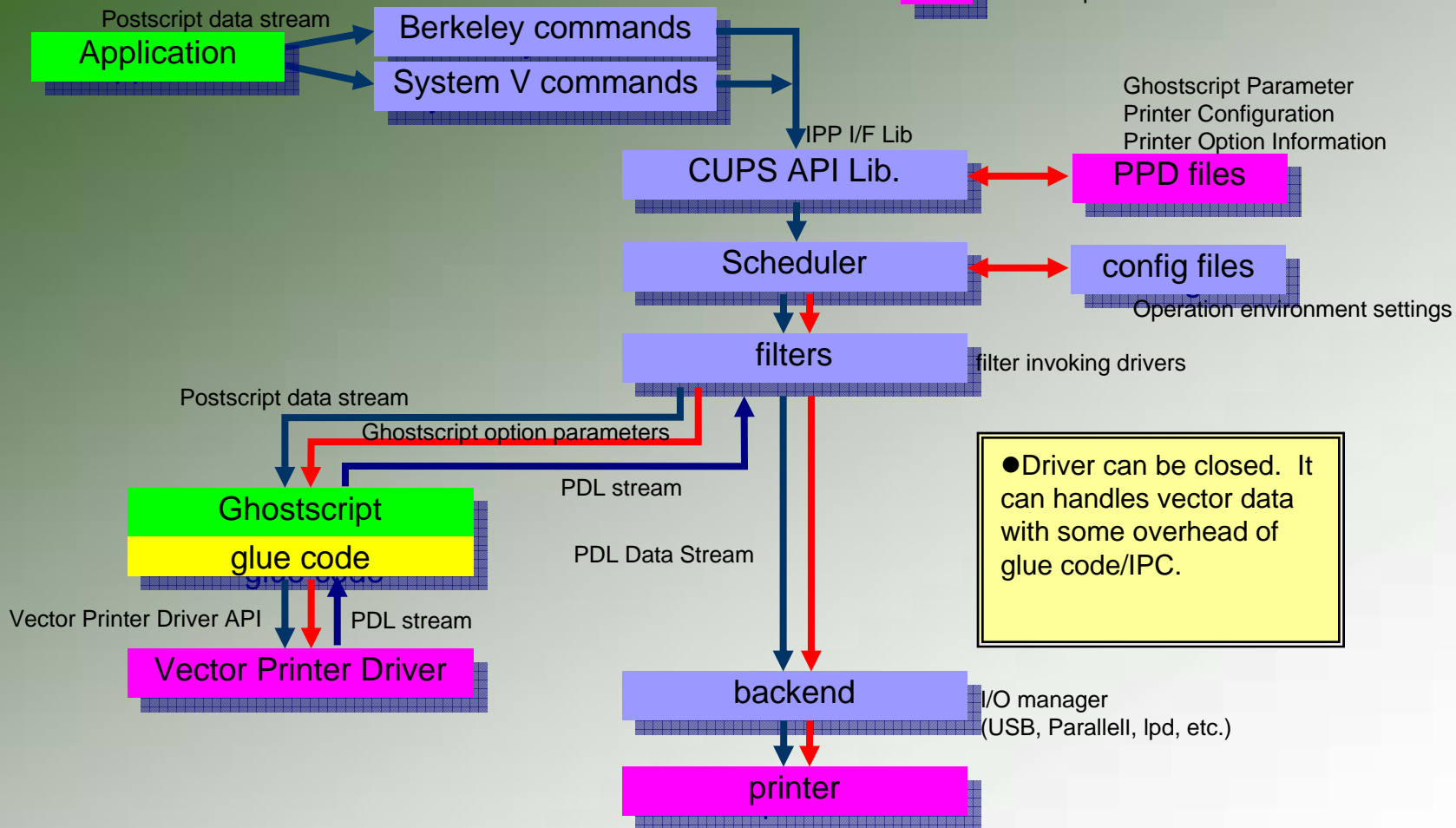
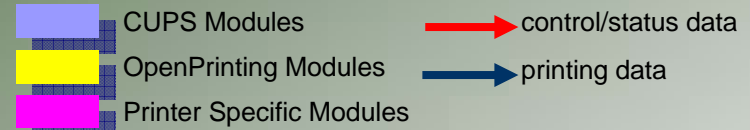


Ghostscript+Raster Printer Driver

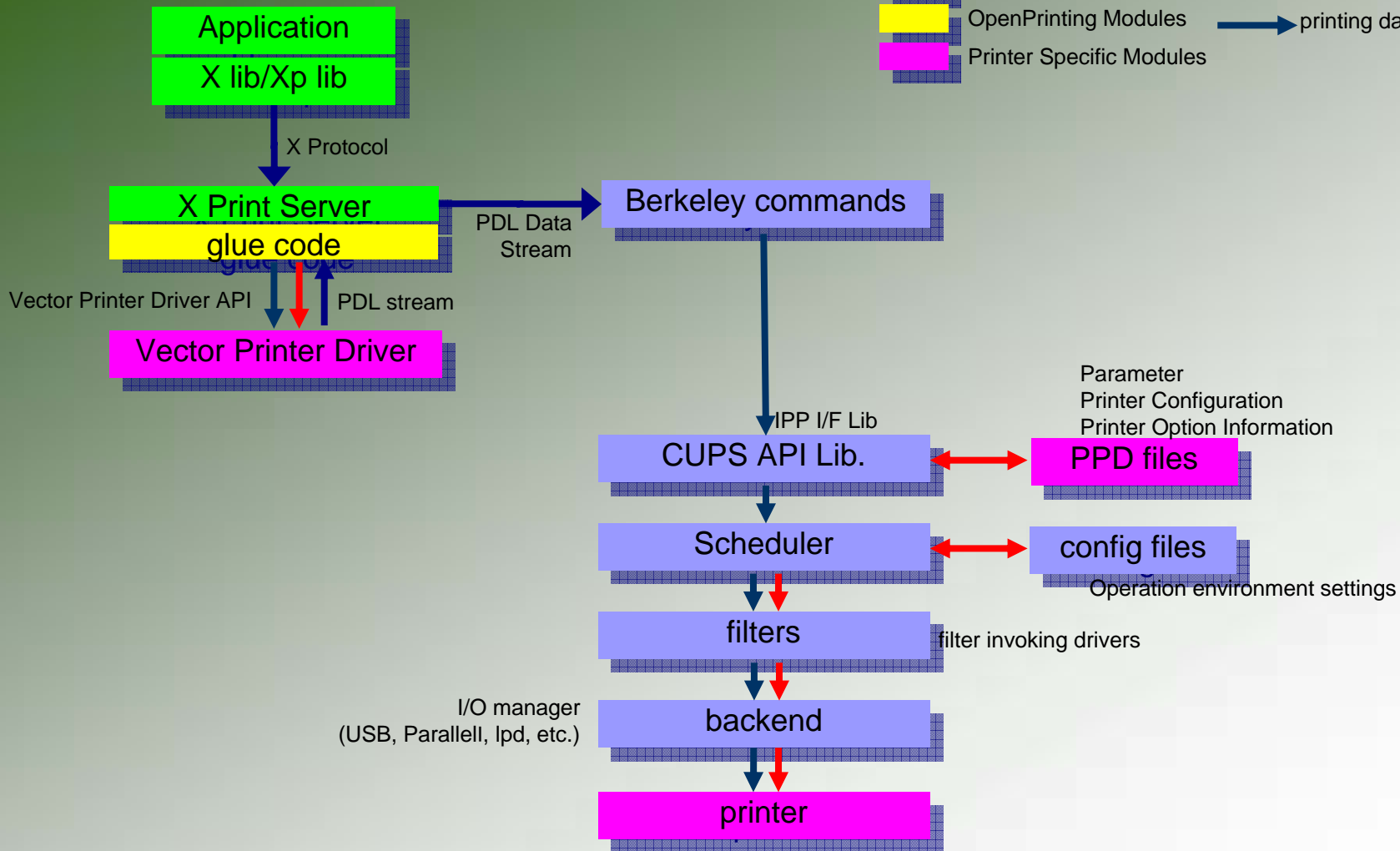
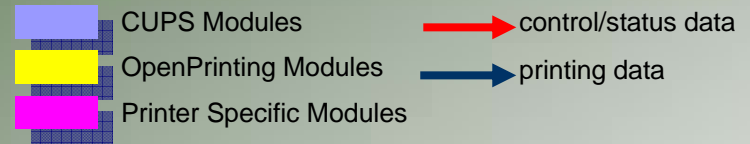




IPA Implementation (1): Ghostscript+Vector Printer Driver



IPA Implementation (2): Xprint+Vector Printer Driver





■ Specification

- ✗ V0.2: <ftp://ftp.pwg.org/pub/pwg/fsg/vector/>
- ✗ V0.3pre (work in progress)
 - ◆ <http://omihara.hp.infoseek.co.jp/unixprint/vector/VectorPrinterDriverAPI-20040906.sxw>

■ Implementation)

- ✗ <https://sourceforge.jp/projects/opfc/>

■ Official Drivers (sites in Japanese)

- ✗ http://cweb.canon.jp/drv-upd/lasershot/drv_linux.html
- ✗ http://www.epkowa.co.jp/linux/dl_OPFC.html

■ IPA Project

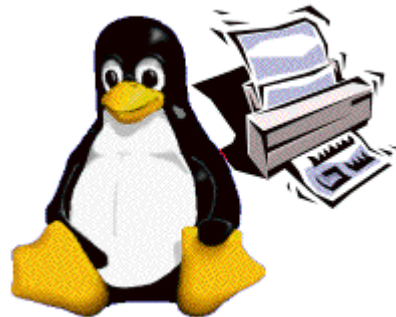
- ✗ http://www.epkowa3.on.arena.ne.jp/OpenPrintingProject/index_e.html





Free
Standards
Group

Printer Driver





Printer Driver & Objectives

✚ Printer Driver API is:

- ✘ A printer driver interface for requesting driver/printer information and accepting/printing print jobs.
 - ◆ Vector API's supporting PDL based printers
 - ◆ Raster API's supporting Raster based printers

✚ Printer Driver API contains:

- ✘ Commands to query/set capabilities
- ✘ Commands to create and control print jobs
- ✘ Vector / Raster transfer commands

✚ Objectives

- ✘ To be a common interface for printing to printers
- ✘ To isolate the application from the details of individual printers
- ✘ To isolate the application from the details of individual PDLs
- ✘ To have printer drivers support a set of common job properties
- ✘ Performance Optimization
 - ◆ Achieve full speed printing
 - ◆ Utilizes graphical acceleration feature supported by printer controllers



Printer Driver API

- Job Control
 - ✗ Open/Close driver
 - ✗ Set Job/Document/Page attributes

- Graphics State Operation
 - ✗ Set attributes for each graphics objects

- Drawing Operations
 - ✗ Path
 - ✗ Text
 - ✗ Bitmap Image
 - ✗ Scanline
 - ✗ Raster Image

- Stream Data (embedded PDL)



Printer Driver API – Details (1)

Printer Context

- ✗ **OpenPrinter()**
 - ◆ Create printer context
 - ◆ Register API entry pointers
 - ◆ Specify file descriptor for data stream
- ✗ **ClosePrinter()**
 - ◆ Closes printer context
 - ◆ Driver releases all resources

Job Control

- ✗ A print job consist of documents.
- ✗ A document consist of pages.
- ✗ **StartJob(), EndJob()**
- ✗ **StartDoc(), EndDoc()**
- ✗ **StartPage(), EndPage()**
- ✗ Job, doc and page attributes are specified by each StartXxx() function.

Query Device Capabilities & Information

- ✗ **QueryDeviceCapability()**
 - ◆ Query if the device can do number-up, duplex, etc.
 - ◆ Information such as media size, media source and etc. which are supported by the device can be retrieved.
- ✗ **QueryDeviceInfo()**
 - ◆ Query current settings of the device.

Graphics State Object Operations

- ✗ Graphics State is managed as GS object
 - ◆ Operation to GS – InitGS, SaveGS, RestoreGS
- ✗ Controls to each items in GS
 - ◆ CTM (Coordinate Translate Matrix)
 - ◆ Color Space
 - ◆ Raster Operation – ROP3
 - ◆ Fill Mode – even/odd or winding
 - ◆ Alpha Constant
 - ◆ Line Style – width, dash/solid, cap, join
 - ◆ Paint Mode – opaque or transparent
 - ◆ Stroke and fill color – brush control
 - ◆ Foreground and background color – solid brush

Path Operations

- ✗ A path is a virtual track object
 - ◆ Will be visible by stroke or fill operations
 - ◆ Will be used to define clip region
- ✗ Lines, rectangles, polygons, arc/pie and bezier are all treated as “path.”
- ✗ Operations:
 - ◆ **NewPath()** – Declare start of a path
 - ◆ **EndPath()** – Declare end of a path
 - ◆ **StrokePath(), FillPath(), StrokeFillPath()** – make visible path
 - ◆ **SetClipPath(), ResetClipPath()** – defines clip region by current path



Printer Driver API – Details (2)

Text Operations

- ✘ Still under investigation...
- ✘ Current DrawBitmapText() will be removed.
- ✘ Text Operations will includes:
 - ◆ Define and Query font metrics
 - ◆ Device Font Utilization
 - ◆ Font Downloading

Raster Image Operations

- ✘ StartRaster(), TransferRasterData(), EndRaster()
- ✘ Set to be extended by Raster Team

Stream Data Operations

- ✘ StartStream(), TransferStreamData(), EndStream()

Bitmap and Scanline Operations

- ✘ Bitmap is a bit oriented image data drawn in rectangle region
 - ◆ DrawImage()
 - ◆ StartDrawImage(), TransferDrawImage(), EndDrawImage()
- ✘ Scanline is a horizontal line defined by start and end point pairs.
 - ◆ Used to draw graphics rendered by renderer
 - ◆ StartScanLine(), ScanLine(), EndScanLine()

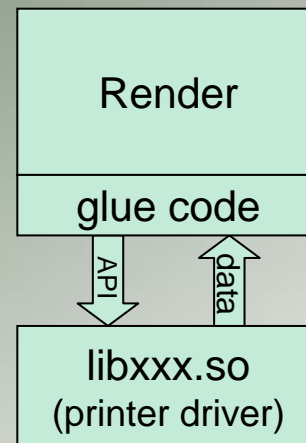


Linking

- Printer driver is provided as a dynamic library.
- Driver can be linked dynamically or via RPC.
 - avoids license problem

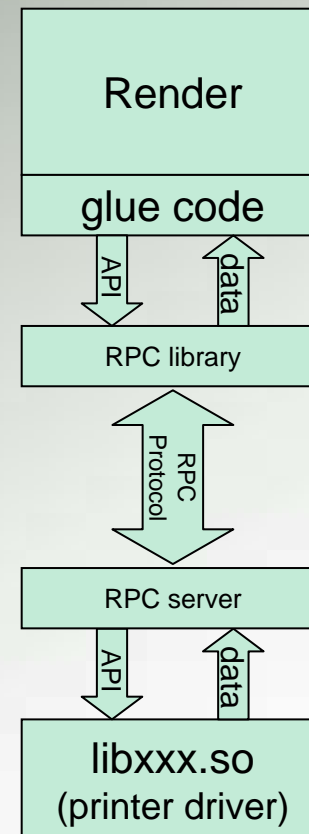
direct linking

R: GPL
D: GPL
or
R: MIT
D: Closed or LGPL



RPC linking

R: any
D: any





PDAPI Raster- Common Job Properties

Features

- ✘ Standardized name for common features
 - ✘ Standardized keys and the values
 - ✘ An extensible paradigm for non-standard features
 - ✘ Coherence across the FSG OpenPrinting model
-
- | | |
|---------------------------------|--------------------------|
| ✘ ColorInput | ✘ PrintQuality |
| ✘ ColorOutput | ✘ Resolution |
| ✘ Copies | ✘ Rotation |
| ✘ Margins | ✘ ScalingType |
| ✘ MediaBackCoating | ✘ ScalingPercentage |
| ✘ MediaColor | ✘ SheetCollate |
| ✘ MediaFrontCoating | ✘ Sides |
| ✘ MediaInputTrayName | ✘ StitchingPosition |
| ✘ MediaSizeName | ✘ StitchingReferenceEdge |
| ✘ MediaType | ✘ StitchingType |
| ✘ MediaUnprintableMargins | ✘ StitchingCount |
| ✘ NumberUp | ✘ StitchingAngle |
| ✘ NumberUpPresentationDirection | ✘ Trimming |
| ✘ OutputBinName | |



PD-Raster Issues / Concerns

- Small number of contributors in this group
- Contributors from across the printing spectrum
- Integration of the vector API into this group
- Reconciliation with other standards
- Expansion into other printing areas
 - ◆ - Device fonts
- Funding for people on this group



PD-Vector Working Group Information

- To subscribe to FSG Vector Printer Driver mailing list:
 - ✘ <http://freestandards.org/mailman/listinfo/printing-japan>
- To post a message to FSG Vector Printer Driver mailing list
 - ✘ printing-driver@freestandards.org
- To view FSG Vector Printer Driver mailing list archives
 - ✘ <http://freestandards.org/mailman/listinfo/printing-japan>
- To find FSG Vector Printer Driver documents
 - ✘ <ftp://ftp.pwg.org/pub/pwg/fsg/vector/>
- Participants
 - ✘ Osamu Mihara (Fuji Xerox Printing Systems Co., Ltd)



PD-Raster Working Group Information

- Weekly FSG Printer Driver conference calls
 - ✘ Mondays at 2:00 PM US Eastern for 1-2 hours
- To subscribe to FSG Printer Driver mailing list:
 - ✘ <http://freestandards.org/mailman/listinfo/printing-driver>
- To post a message to FSG Printer Driver mailing list
 - ✘ printing-driver@freestandards.org
- To view FSG Printer Driver mailing list archives
 - ✘ <http://freestandards.org/mailman/listinfo/printing-driver>
- To find FSG Printer Driver documents
 - ✘ <ftp://ftp.pwg.org/pub/pwg/fsg/driver/>
- Participants
 - ✘ Mark Hamzy (IBM) – chair
 - ✘ Till Kamppeter (MandrakeSoft)
 - ✘ Glen Petrie (Epson)



Thanks
from the
Printer Driver
Team





Thanks
from the
Status Monitoring
Team

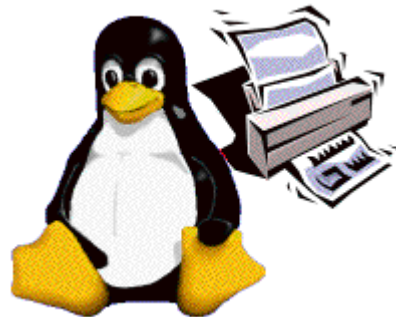




**Free
Standards
Group**

PCM Study & Status Monitoring Interface

TORATANI Yasumasa
toratani.yasumasa@canon.co.jp
OpenPrinting WG Japan/Asia
Canon Inc.
15-17 November 2004





- PCM Plug-in Interface
 - ✗ Backgrounds
 - ✗ Requirements
 - ✗ API Design

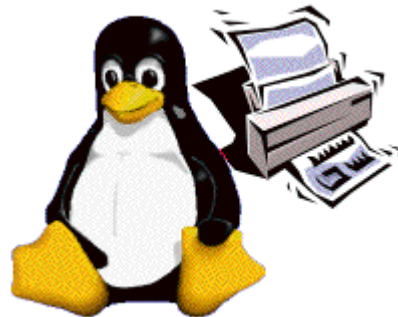
- Status Monitoring Interface
 - ✗ Just an idea

- Issues/Concerns, Next and Info.



**Free
Standards
Group**

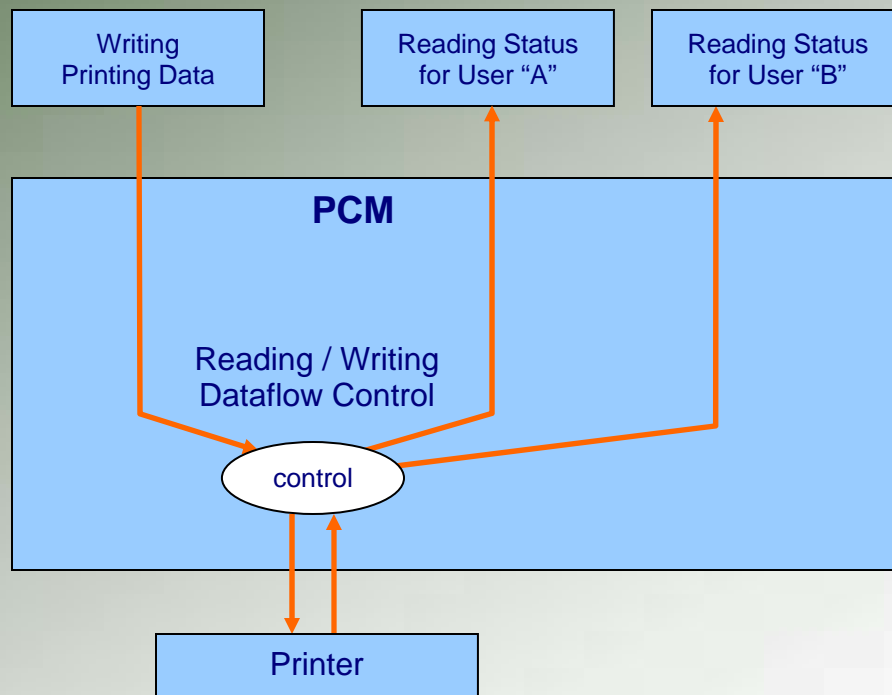
PCM Plug-in Interface






Backgrounds

- PCM provides the multi-channel interface for;
 - Writing a printing data to the printer.
 - Reading a printer status data from the printer.





Backgrounds (Cont.)

- Needs to control the multi-channel data flow.
 - ✘ Who controls?
 - Each printer/protocol has its own commands and procedures.
 - ✘ Who deals with the printer/protocol dependencies?
- 
- PCM Should be separated into two layers;
 - ✘ Core module providing the multi-channel functionalities.
 - ✘ **Plug-in** providing each printer/protocol dependent functionalities.



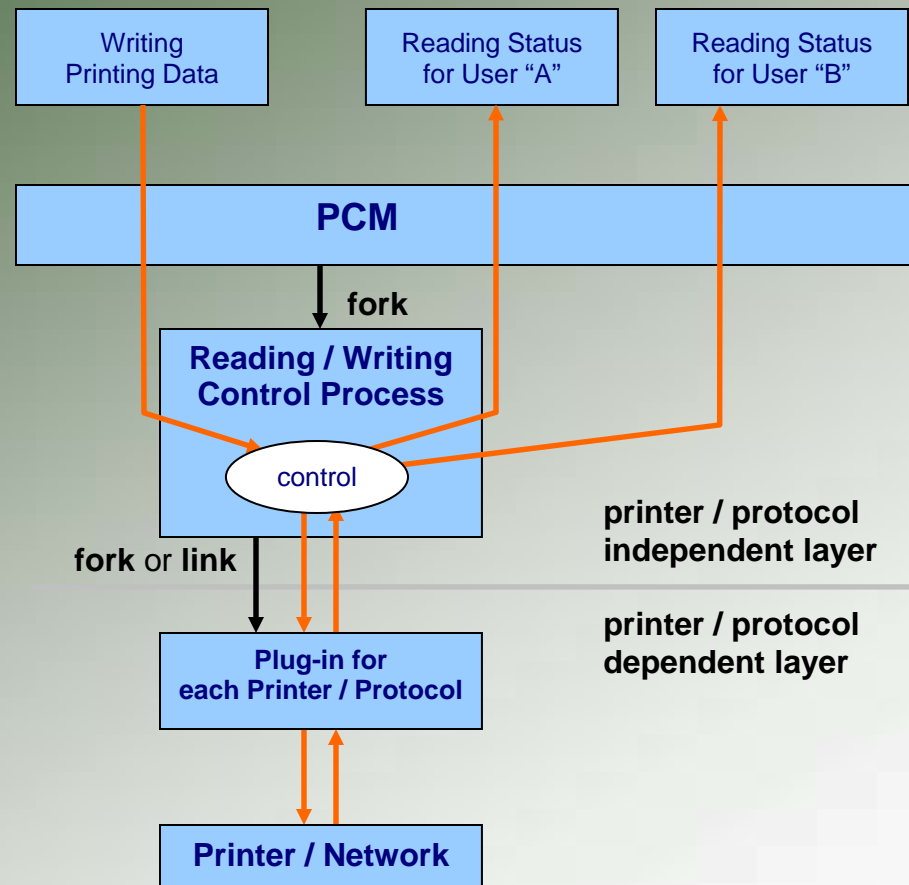
Two Layers Arch.

Reading / Writing Control Process

- ◆ Forked by PCM for each printer.
- ◆ Manages the dataflow due to;
 - Data writing request by PCM.
 - Data reading request by PCM.
 - Data writing ready from Plug-in.
 - Data reading ready from Plug-in.

Plug-in for each Printer / Protocol

- ◆ Manages the printer / protocol dependent;
 - Packet control.
 - Timing Control for reading / writing.
 - Data translation.





Two Layers Arch. by threads

Reading / Writing Control Process

Writing Thread

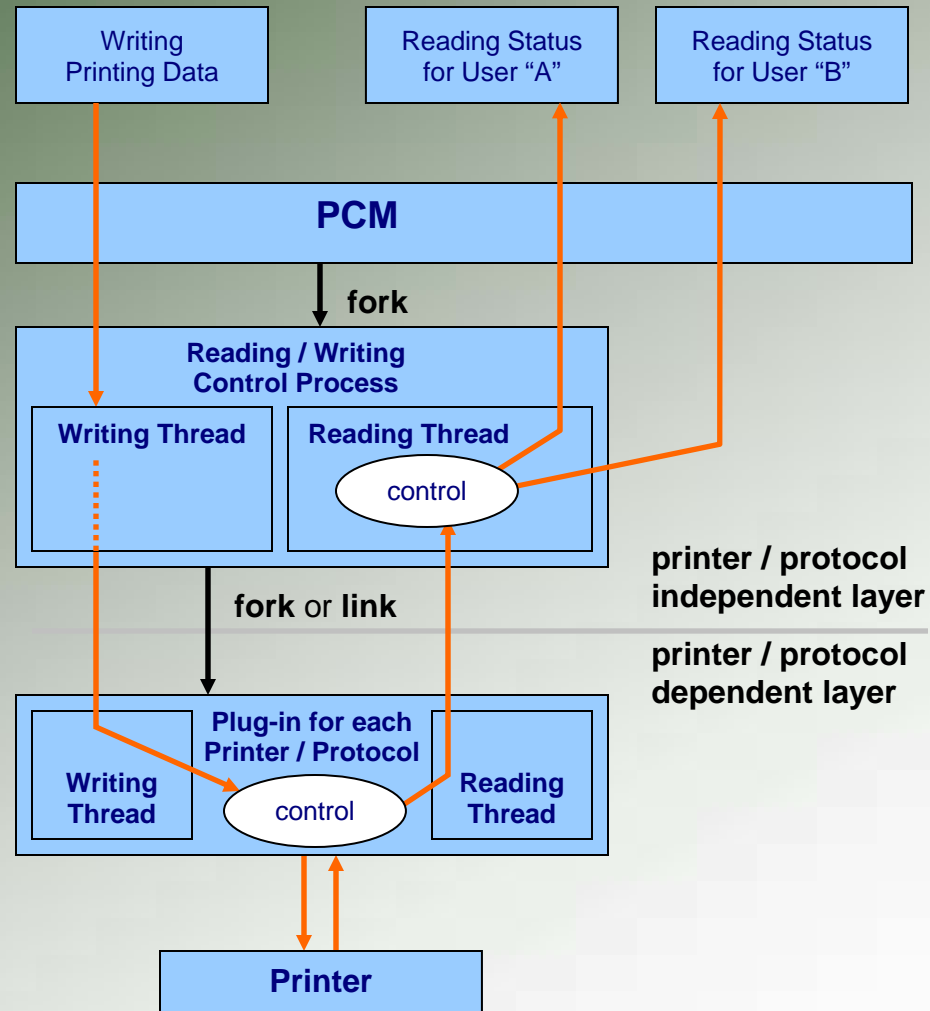
- ◆ Manages the dataflow due to;
- Data writing request by PCM.
- Data writing ready from Plug-in.

Reading Thread

- ◆ Manages the dataflow due to;
- Data reading request by PCM.
- Status reading ready from Plug-in.

Plug-in for each Printer / Protocol

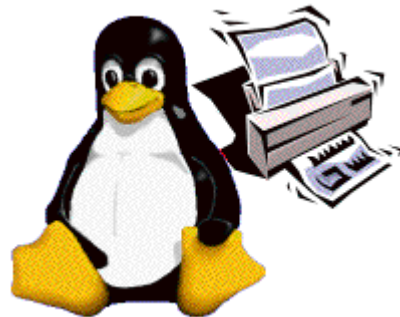
- ◆ Manages the printer / protocol dependent;
- Packet control.
- Timing Control for reading / writing.
- Data translation.





**Free
Standards
Group**

Requirements for Plug-in





Basic Requirements

❏ Plug-in must;

A-1) Provide the API for;

- ❖ Writing a printing data to the printer.
- ❖ Reading a printer status data from the printer.

A-2) Conceal the printer/protocol dependencies.

- ❖ PCM can deal with all the printers/protocols (e.g. IPP, etc..) as an unified object via each Plug-in.

A-3) Provide the API that can be applied on both;

- ❖ Big memory system.
- ❖ Small memory system.



Port Control Requirements

❏ Plug-in must;

B-1) Conceal the port control procedures based on each printer/network protocol dependencies.

- ◆ Open / Close a USB port, etc...
- ◆ Open / Close an IPP port etc...

❏ Port must;

B-2) Be specified by the generic device/network protocol identifier.

- ◆ URI ?



Data Reading Requirements

■ Plug-in must;

c-1) Provide a generic API for sending back the variable length printer status.

- ◆ Printer status data length depends on each printer model and conditions of the printer.
- ◆ Plug-in may separate a single very long printer status data into a set of several short data for sending them back into a short restricted buffer prepared by the upper layer.

c-2) Provide the API to know the top of the printer status data and/or to reset the reading position of it.

- ◆ The upper layer doesn't know what the status data includes(binary, XML, etc...), but needs to know the top of it.
- ◆ Reading a printer status data may be interrupted by a signal.



Data Reading Requirements (Cont.)

✚ Plug-in must;

c-3) Keep the printer status data consistent until all the status data are read by the upper layer.

- ✚ Plug-in needs a trigger to allow for reading a printer status from the printer/network port.

c-4) Provide the API which gives a trigger to the Plug-in for dealing with reading a printer status data and writing a printing data exclusively.

- ✚ Some printers/protocols have to deal with reading a printer status data and writing a printing data exclusively, and the Plug-in needs a trigger for doing it.



Data Writing Requirements

❏ Plug-in must;

D-1) Provide the API which gives a trigger to the Plug-in for dealing with reading the printer status data and writing the printing data exclusively.

- ◆ Same as C-4) of the Data Reading Requirements.
- ◆ Writing a printing data may be interrupted by a signal.



Dataflow Control Requirements

❏ Plug-in must;

E-1) Let the upper layer know;

- ◆ If the Plug-in is ready to receive a printing data.
- ◆ If the Plug-in is ready to send back a printer status data.

❏ Plug-in should;

E-2) Provide the API being not affected by OS platform dependencies.

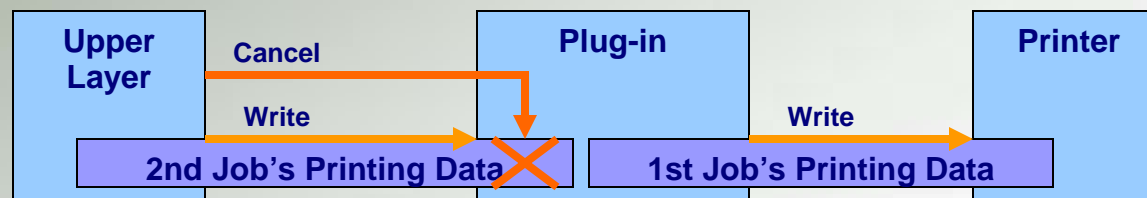


Printer Control Requirements

■ Plug-in must;

F-1) Provide the API to cancel a printing.

- ◆ The upper layer keeps alive the Plug-in for obtaining the printer status continuously across several printing jobs.
- ◆ Each printing job's printing data might be specified by an identifier for the cancellation. (TBD)
 - ◆ Example) In case of the upper layer sends two job's printing data to the Plug-in. After the upper layer finished to send the first job's printing data to the Plug-in, the upper layer can start to send the second job's printing data while the Plug-in may send the first job's printing data to the port. During this period, the first job has not been finished, and if the user wants to cancel the second job after noticed the mistake, the upper layer needs an identifier to cancel the second job's data buffering, or cancel the first job.





Printer Control Requirements (Cont.)

❏ Plug-in might (TBD);

F-2) Provide the API for the common control functions.

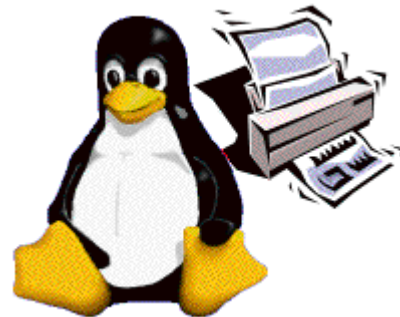
- ❖ “Pause”, “Resume” for local port printers as well as IPP ?
 - IPP needs a job identifier to control a job.

F-3) Provide the API for tracking printing jobs.

- ❖ Plug-in for IPP will be provided, while IPP provides job attributes.
- ❖ If PCM is the “only” interface to access a printer, PCM might provide the interface for tracking printing jobs in a printer to satisfy some applications needs.



Plug-in API Design





Object / Port Control

Object Control

New

- Conceal the printer/protocol dependencies in the created object.
- Object keeps the URI specified by the upper layer until destroyed.

Destroy

- Destroy the object.

Port Control

Open

- Open the printer / network port specified by the object.

Close

- Close the port / network port specified by the object.



Data Reading

StartRead

- Declare the start of reading a printer status.
- Reset the reading position of the printer status data in the object.

Read

- Read the specified byte length of a printer status data from the object, and store them in the buffer given by the upper layer.
- Returns the status data byte length read from the object, or zero when no status data remains in the object.

* The data format of the printer status is out of scope of this API.

EndRead

- Declare the end of reading a printer status.



Data Writing

StartWrite

- Declare the start of writing a printing data.

Write

- Write the specified byte length of a printing data in the buffer given by the upper layer to the object.
- Returns the printing data byte length written to the object, or zero when no printing data written to the object.

EndWrite

- Declare the end of writing a printing data.



Dataflow Control

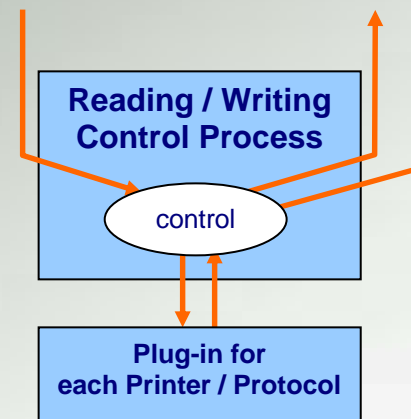
GetReadFD

- Obtain the reading file descriptor for select() use.
- The upper layer know by select() if the Plug-in is ready to send back a printer status data.

GetWriteFD

- Obtain the writing file descriptor for select() use.
- The upper layer know by select() if the Plug-in is ready to receive a printing data.

The upper layer deals with all the reading/writing request by select().





Printer Control

StartJobData

- Declare the beginning of the printing data of each job to the object.

CancelJobData

- Cancel the printing data specified by the job identifier.

EndJobData

- Declare the end of the printing data to the object.

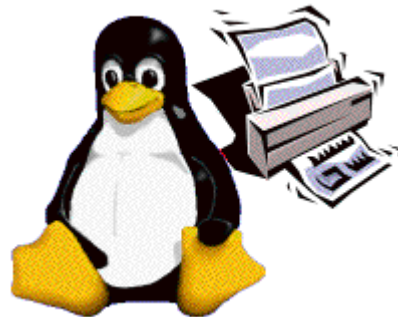
ExtCtrl

- Execute the extended printer control, such as “Pause”, “Resume”...



**Free
Standards
Group**

Status Monitoring





✚ Requirements

- ✚ Status Monitoring Interface must;
Translate the printer status data read from each Plug-in into the generic format data that does not depend on each printer/protocol.

- ✚ Generic format is XML? IPP? other?



Just an Idea (Cont.)

■ API Design

New

- Conceal the printer/protocol dependencies in the created object.
- Object keeps the “language” and “character encoding” specified by the caller (module which calls this API) until destroyed.

StartTranslate, Translate, EndTranslate

- Translate the printer status data given by the caller to the generic format data, and store it into the buffer given by the caller.

Destroy

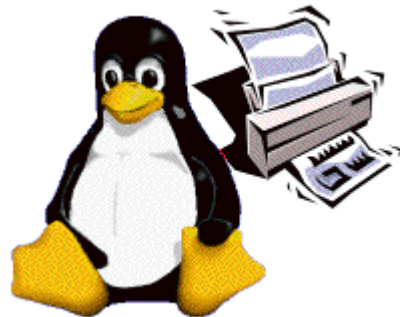
- Destroy the object.





**Free
Standards
Group**

Issues/Concerns, Next and Info.



Issues/Concerns, Next



■ Plug-in Interface

- ✗ Need to collect the requirements for handling events.

■ Status Monitoring Interface

- ✗ Need to collect the requirements for the Status Monitoring functionalities.

■ Next

- ✗ Define the draft Plug-in API.
- ✗ Define the draft Status Monitoring API.



Contributors

TORATANI Yasumasa

Osamu MIHARA

Ide Kentaro

Nomura Kazuo

KANJO Hidenori

YOSHIDA Mikio

Shinpei KITAYAMA

YAMAGISHI Toshihiro

Hisao NAKAMURA

Koji OTANI

Canon Inc.

FUJI XEROX Printing Systems Co. Ltd.

SEIKO EPSON CORPORATION

EPSON SOFTWARE DEVELOPMENT LABORATORY, INC.

BBR INC.

BBR INC.

EPSON KOWA CORPORATION

Turbolinux, Inc.

E&D

AXE





The End

Thanks from the

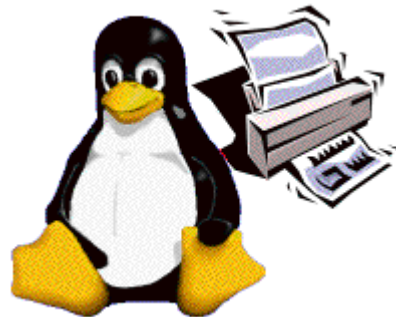
*FSG | OpenPrinting
Working Groups*





Free
Standards
Group

Print Channel Monitor





Print Channel Monitor Objective

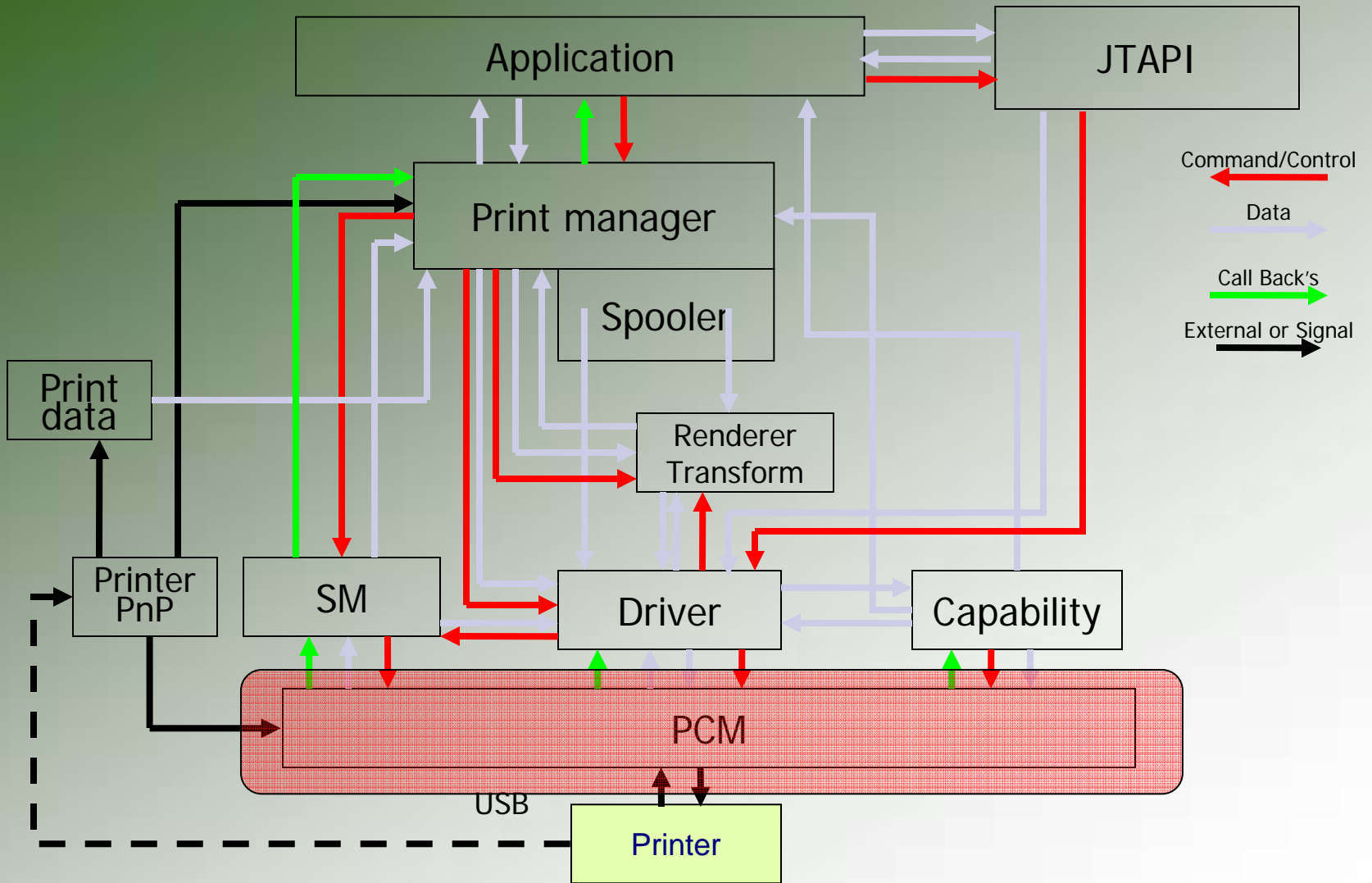
- Print Channel Monitor (PCM) will wrap the difference of physical layer
 - ✗ Upper layer do not need to care about the difference of physical layer like USB, Parallel, Ethernet.
 - ✗ Wrap the difference of transaction protocol used by printer venders.

- PCM will focus on communication layer
 - ✗ Job control and device management are left to upper layer.



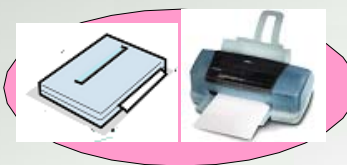
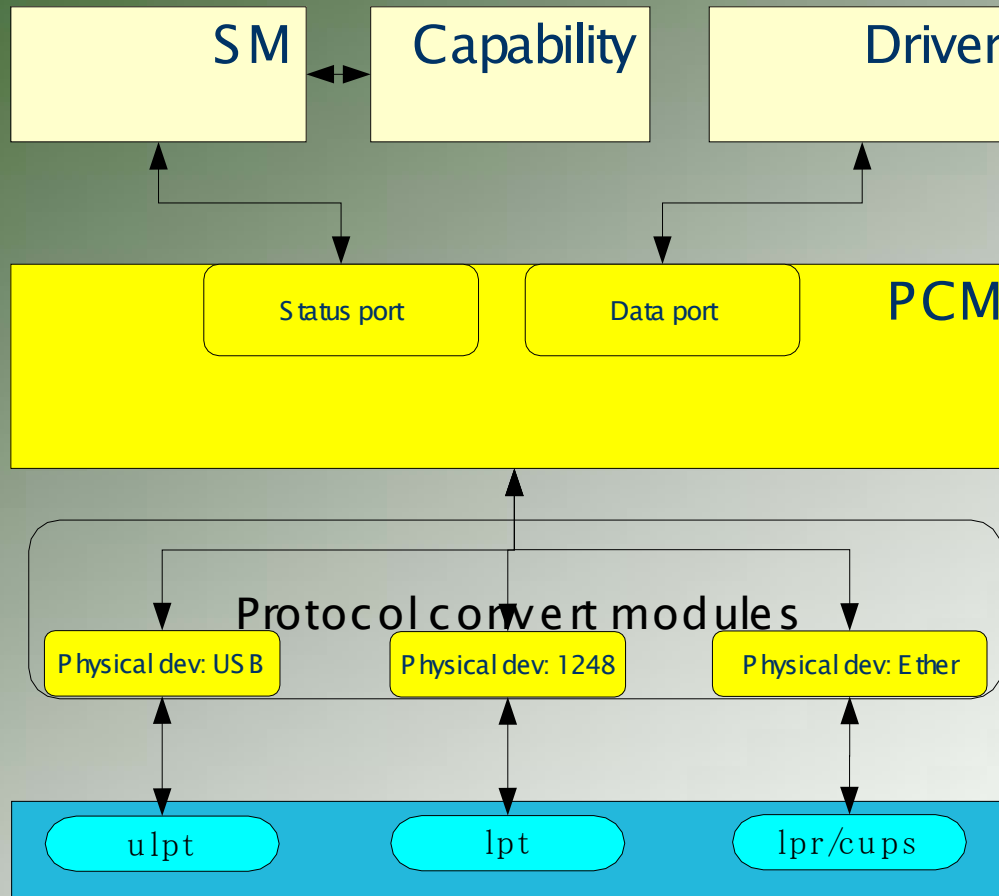


PCM outline





PCM outline





■ Draft – working now!

- 2004/5/31 Pre-draft of the PCM specifications.
(written only the main point.)
- 6/ F Open to working group.
- 6/ F Start discussion and writing the draft.

■ Discussion about PCM

- ✗ Printing-Japan ML
- ✗ Face to Face meeting – Monthly

■ Milestone

- ✗ 2004/8/27 Create a model/architecture
- ✗ 2005/1/30 Release 1.0 specification

■ To-Do

- ✗ Consider the other existing Printing Systems so PCM could be easily adopted
- ✗ Consider the future extension
 - ◆ Support new I/F or new device like MFP





PCM Working Group Information

- To subscribe to FSG PCM mailing list:
 - ✗ <http://freestandards.org/mailman/listinfo/printing-japan>
- To post a message to FSG PCM mailing list
 - ✗ printing-japan@freestandards.org
- To view FSG PCM mailing list archives
 - ✗ <http://freestandards.org/mailman/listinfo/printing-japan>
- To find FSG PCM documents
 - ✗ To be determined
- Participants
 - ✗ Kentaro Ide (Seiko Epson Corp)
 - ✗ Shinpei Kitayama (Epson Kowa Corp.)
 - ✗ Fumio Nagasaka (Seiko Epson Corp.)
 - ✗ Kazuo Nomura (Seiko Epson Corp.)
 - ✗ Koji Otani (AXE Inc)





Thanks
from the
Print Channel
Monitoring
Team

