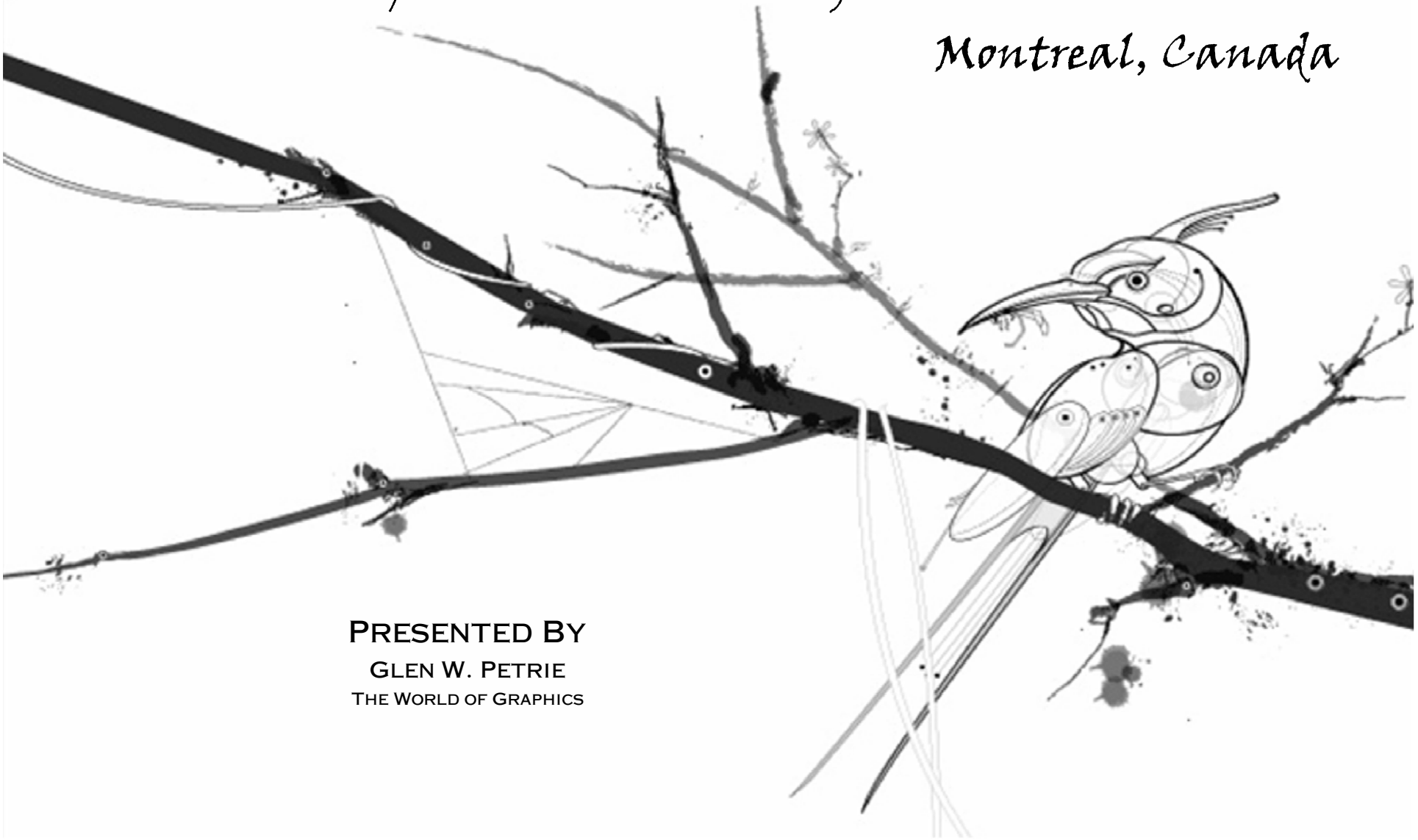
Linux Foundation: Open Printing

Open Summit Meeting, Fall 2007

Montreal, Canada



Presented By

Glen W. Petrie

The World of Graphics

Agenda

- Introduction
  - Me to Open Printing
- Printing Environments
  - Production to Embedded
- Coherence
  - Environments to Software
- Scalability
  - Environments to Software
- Models
  - Production to Embedded
- Software
  - Basic to Core to Thin-Thread to Solution
- Where are we …
  - Architecture to Thin-Threads
- Where are we going …
  - Thin-Threads to Solutions

Introduction

- About Me
- Overview
  - Open Printing History
  - Open Printing Application Programming Interfaces
  - Going Forward

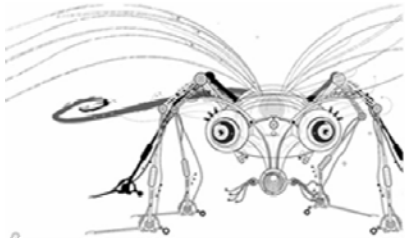
Introduction: About Me

- Glen W. Petrie
  - Printing/Print Work Experience
    - Epson Portland, Inc.: Senior Software Architecture / Principle Engineer
      - Linux, WinCE and Embedded Core Drivers and Solutions for Consumer Inkjet Printers
    - Xerox PARC: Principle Scientist/Engineering
      - Production and Large Office Printing Architectures.
      - Data Glyph Technology Architecture, Design and Development.
        - » Production, Office and Embedded Solutions
  - Open Printing Background
    - Was there on the first day in San Jose on Oct 25-26, 2001
    - Member of the Open Printing
      - Steering Committee,
      - Architecture Team,
      - Job Ticketing Working Group and
      - Raster Driver Working Group
    - Currently the Designer and Developer for the
      - Open Printing Embedded Print Solution

Introduction: Open Printing

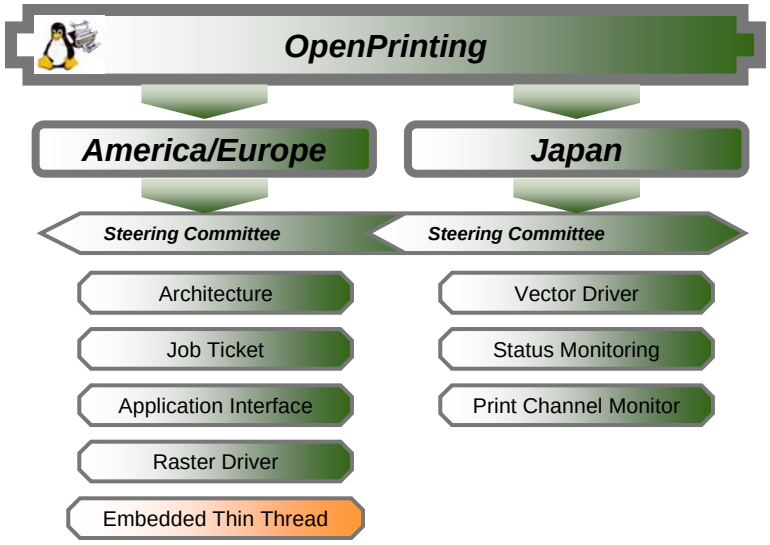
- The first Print Summit Meeting in San Jose, California on Oct 25-26, 2001

- Objective
  - "Standardizing on a Scalable Print Environment in Linux."



- Mission Statement

  The mission of the Open-Printing is to *develop and promote a set of standards* that will address the needs of *embedded to desktop to enterprise-ready printing;* including *management, reliability, security, scalability, printer feature access and network accessibility.*



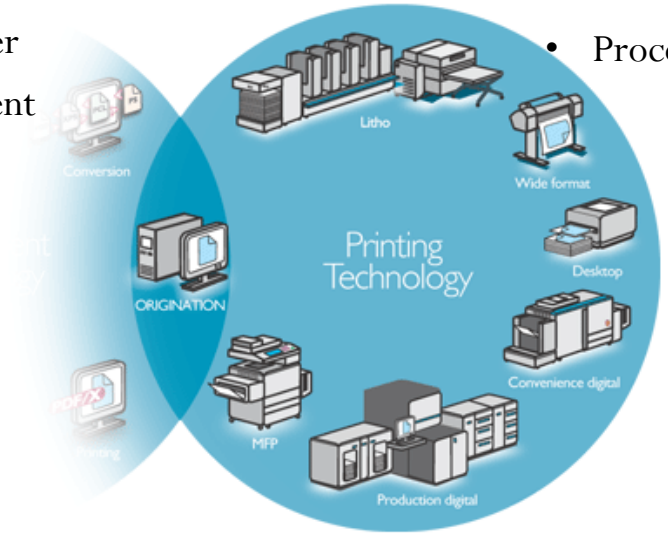
Environments

- We identify four principle Printing Environments
  - Production Printing
  - Office Printing
  - Home Printing
  - Embedded Printing

- Within each of these we can …
  - … define a continuum of sub-environments
  - … define distinct niche sub-environments

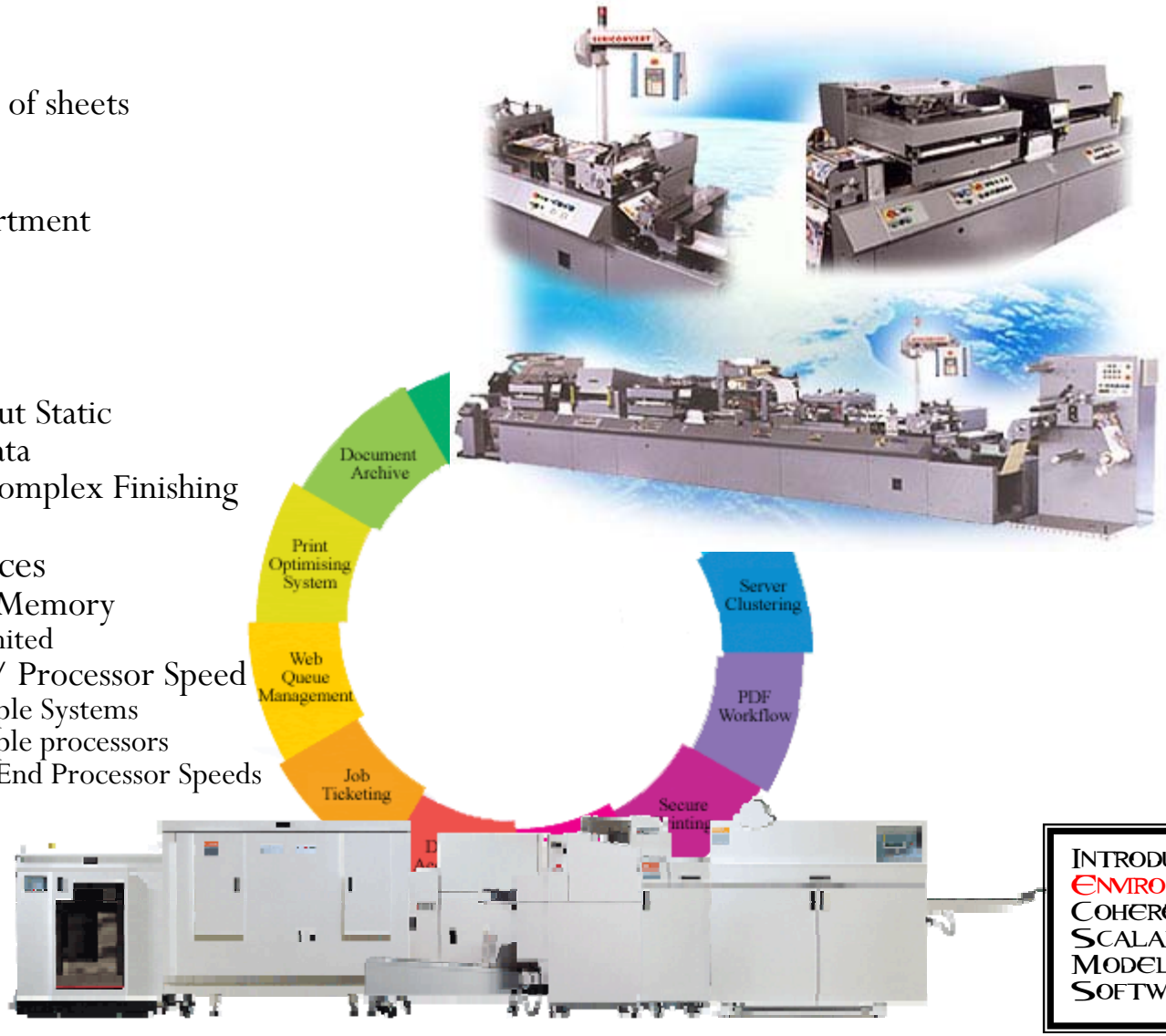
Environment Factors

- What factors distinguish Printing Environments
  - Print Volume
    - 1 to 10's of sheets
    - 10's to 100's of sheets
    - 1000's of sheets
  - Print Location
    - Attached Printer
    - Network Printer
    - Print Department
    - Print Shop
  - Print Job Type
    - Simple
    - Complex but Static
    - Variable Data
  - System Resources
    - Run-Time Memory
    - Processor Speed



Production Environment

- Factors
  - Print Volume
    - 1 to 1000's of sheets
  - Print Location
    - Print Department
    - Print Shop
  - Print Job Type
    - Simple
    - Complex but Static
    - Variable Data
    - Simple / Complex Finishing
  - System Resources
    - Run-Time Memory
      - Unlimited
    - Processors/ Processor Speed
      - Multiple Systems
      - Multiple processors
      - High End Processor Speeds



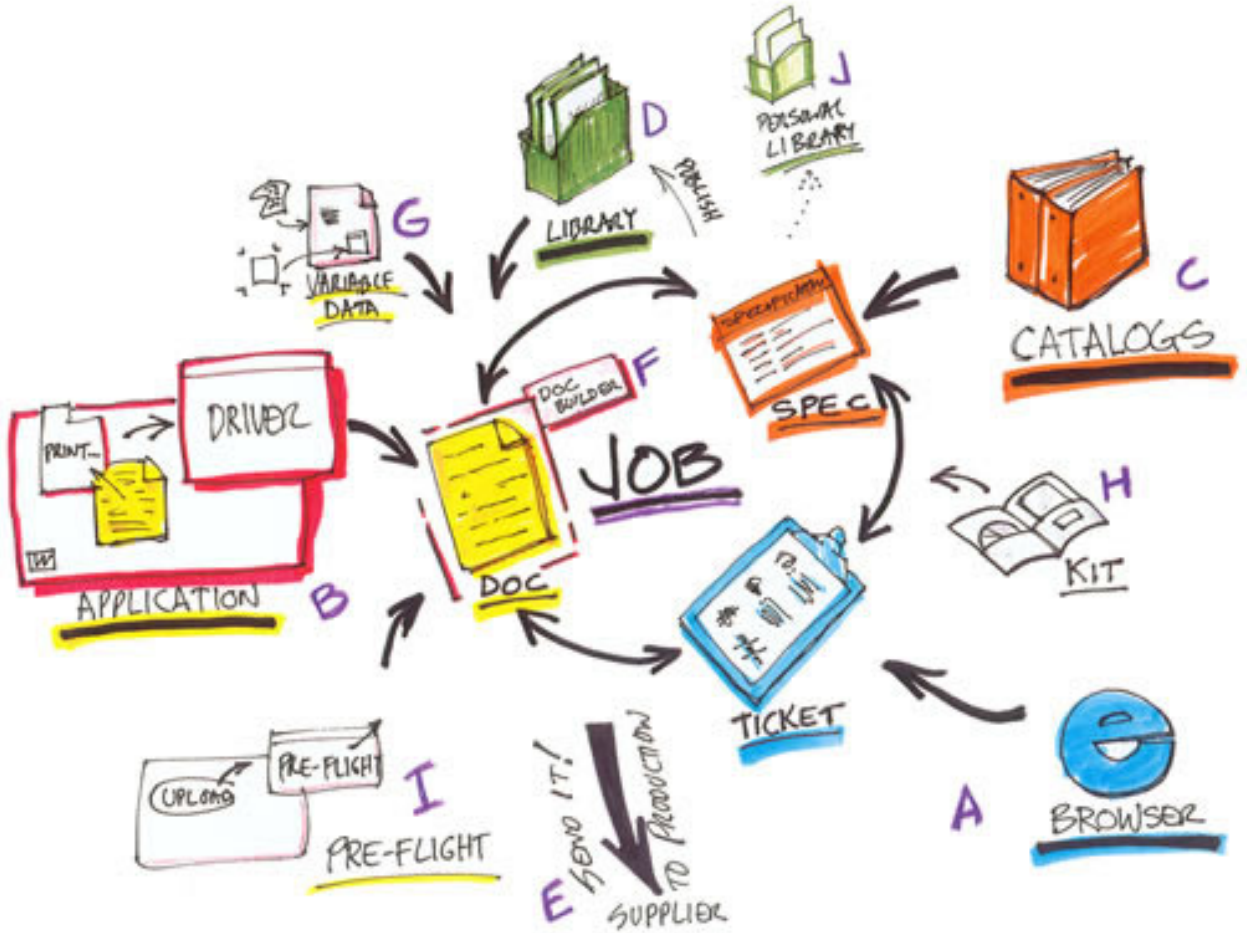
Production Environment

- The Real World !!!

INTRODUCTION
ENVIRONMENTS
COHERENCE
SCALABILITY
MODELS
SOFTWARE

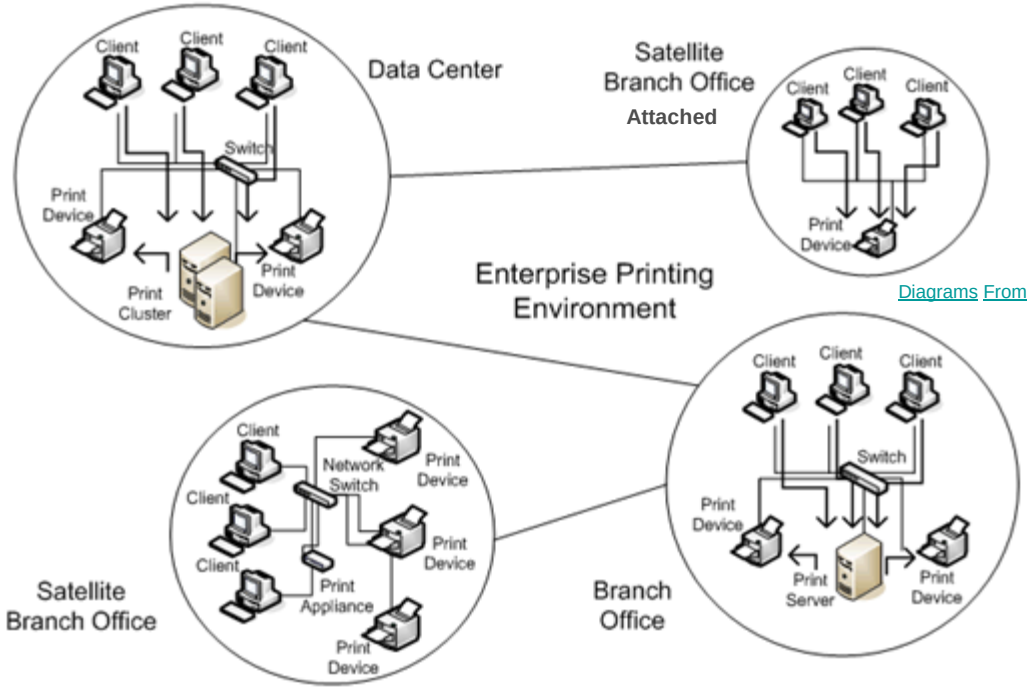
Production Environment

- The Real Process !!!

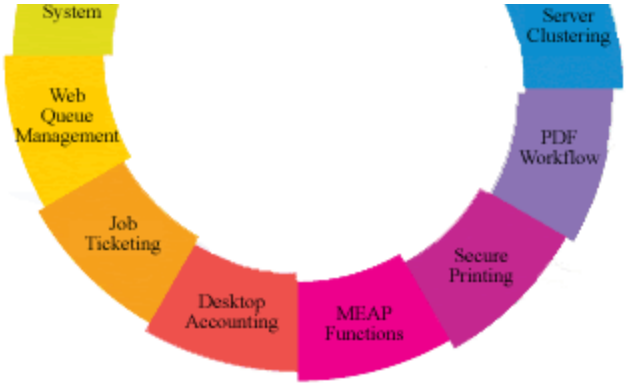


Office Environment

- Factors
  - Print Volume
    - 1 to 100's of sheets
  - Print Location
    - Network Printer
    - Attached Printer
  - Print Job Type
    - Simple
    - Complex but Static
    - Simple Finishing
  - System Resources
    - Run-Time Memory
      - Not Constrained
    - Processors/ Processor Speed
      - Single or More Systems
      - Single or More Processor
      - Not Speed Constrained



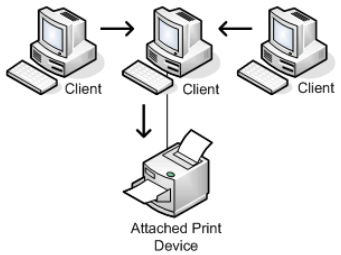
Diagrams From



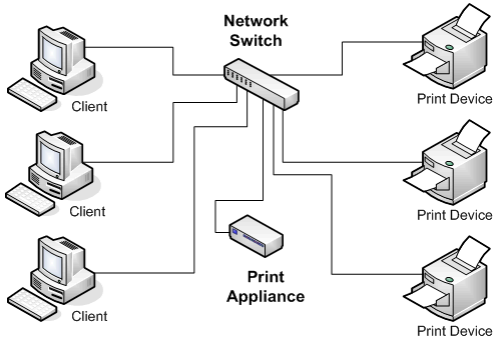
INTRODUCTION
ENVIRONMENTS
COHERENCE
SCALABILITY
MODELS
SOFTWARE

Home Environment

- Factors
  - Print Volume
    - 1 to 10's of sheets
  - Print Location
    - Home Network Printer
    - Attached Printer
  - Print Job Type
    - Simple
  - System Resources
    - Run-Time Memory
      - Not Constrained
    - Processors/ Processor Speed
      - Single Systems
      - Single Processor
      - Not Speed Constrained

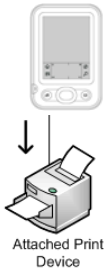


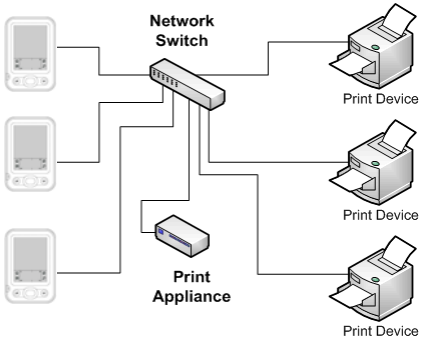
Diagrams From



Embedded/Handheld Environment

- Factors
  - Print Volume
    - 1 to 10 of sheets
  - Print Location
    - Network Printer
    - Attached Printer
  - Print Job Type
    - Simple
  - System Resources
    - Run-Time Memory
      - Less than 1 MiB
    - Processors/ Processor Speed
      - Single Systems
      - Single Processor
      - Less than 500 MHz





Coherence

- Environment Level
  - Means the Users in all environments can (/will/shall/should??) have the same experience.
  - Difference are mostly artificial
    - Production can request the printing a single sheet
      - Print a missing or damaged sheet
    - Handheld can request the printing of a 100 copies
      - Kinko's prints 100 set of a presentation downloaded from a customer PDA
        - » Who generated the production job-ticket !!! Could (should?) the PDA do that? - *Interesting*
  - Use a Scaleable Approach
- Software Level
  - User Level
    - Print Dialog - *Common*
    - Print Attributes - *Common representation and terminology*
  - Developer Level
    - Print Attributes - *Common representation and terminology*
    - Application Programming Interface (API) - *Design, Format, Calls, Error, etc.*
    - Code Module - *Coding Style, Coding Structure, Variable Typing, etc.*
    - Extension: Planning for Change - *Vendor, Code, Attributes, Modules*

Coherence - Why

- User want consistency
  - "Printing should just work" <<< *User major activity is not printing, but they need it*
  - "Printing is different here!" <<< *User diverted from original task*
  - "That not how I wanted the print to look! – What did I do different from before!?#$?%?! <<< *Now What!!!*
- Application Developers need consistency
  - "Printing should just work" <<< *Developer Application major function is not printing, but they support it*
  - "Printing is different here!" <<< *Developer needs separate print function for individual systems*
  - "That not how I wanted the print to look! – What did I do different from before!?#$?%?!" <<< *Now What !!!*
- Printer Driver & OS Developers are expecting consistency
  - "The print attributes are defined differently here!" <<< *Developer creates hash table for attributes*
  - "The attributes has different units and three additional value!" <<< *Developer creates a super set with hash table*
  - "The objects, object data and even data types are different between the two API's <<< *!#$%$% Now What !!!*
  - On & On & On & On . . . .

Coherence Needs (1)

- A Single Dictionary
  - Independent a of Environment, Print/Solution Vendor, Operating System & Application.
  - Defining Terminology, Representation, Relationships, Dependencies &, where applicable, Mathematics
  - Defining Acronyms and Abbreviations
  - Defining the Code Level Variable, Object(Struct) Membership, Range & Scope
    - Use the OpenPrinting Job Ticket Objects & Enums as the Core / Start
- Common / Extensible Print Dialog
  - Being worked on …
  - Provides for both GUI and GUI-less API's
  - Scaleable down to Resource Limited Embedded/Handheld Solutions

Coherence Needs (2)

- Software Level
  - Application Programming Interface (API)
    - Types: Static Link Library, Dynamic Link Library, Remote Processor Call, Other ??
    - Base API's: opInit_*foo*, opProc_*foo*, opRelease_*foo*
  - Base Code Modules
    - Base Headers
      - Base Types (OP_INT8, OP_INT32, OP_CHAR, etc)
      - Base Objects (structs) (OP_RECT, OP_POINT, etc)
      - Base Errors (OP_ERROR_NONE, OP_ERROR_MEMORY_ALLOCATION, OP_ERROR_INVALID_ARG, etc)
  - General Code Module
    - Coding Style – *Pick one and stay with it! – See next page*
    - Coding Structure – *Pick one and stay with it! – See next page*
  - Extension: Planning for Change - *Vendor, Code, Attributes, Modules*

Coherence Needs (3)

```
/************************************|************************************/
/*                                                                  API-150010    */
/* Module     : opInit_JTLib                                                     */
/* Description: Initializes the JTAPI library and return context structure    */
/*              used by this JTAPI library instance.                          */
/*    param >>: none                                                          */
/* << param   : none.                                                         */
/*                                                                            */
/************************************|************************************/
/*                                                                            */
/* Revision Log                                                               */
/*      Date             Who                                                  */
/*      2007.08.07       Glen W. Petrie                                       */
/*    # Original code                                                        */
/*                                                                            */
/************************************|************************************/
void opInit_JTLib (

        void

) {
/*** Declaration of Local Routine Variables                                */

/*** API Identifier Information for this Routine                           */
      OP_API_REG(150010, "opInit_JTLib");

/*** Debug Flow Information for this Routine                               */
    opDebugFlowIn("opInit_JTLib");

/*** Initialization of Local Routine Variables                             */

/*** Initialize Application global variables                               */
    memset(   &theJob, 0, sizeof(OP_JOB    ));
    memset(&theJobTkt, 0, sizeof(OP_JOB_TKT));
    theJobTkt.jobId = theJobId;

/*** Return to Caller                                                      */
    OP_SET_ERR(OP_ERROR_NONE);
    opDebugFlowOut("opInit_JTLib");
    return;
}
```

- Fixed Column Width
- Fixed Routine Header
- Fixed Common Comments
- Fixed Debug/Flow Statements
- etc.

The intent is not control but enable a method for rapid development & consistent development benefiting users and developers.

Scalability

- Environment Level
  - Users in all environments can (/will/shall/should??) have the same experience.
  - Limitation defined by Available Features and Capabilities.
- Software Level
  - User Level
    - Print Dialog – *Common with feature and capability factors*
  - Developer Level
    - Encoding
      - API parametrics … or
      - Printer/Printing capabilities … or
      - Attribute properties …
        - » *… as strings for XML based or resource rich environments*
        - » *… as constants for resource limited environments*

which is "best";
neither and both;
it depends upon environment
and solution space

    - Features and Capabilities
      - The scope, the fidelity and the inclusion based on resources and not necessarily environment !
    - Extension: Planning for Change
      - The scope, the fidelity and the inclusion based on resources and not necessarily environment !

Models (1)

- Architectural Reference Model *(2006.04.10)*