



**LOFTWARE**  
ENTERPRISE LABELING SOLUTIONS

# **Loftware Print Server**

## **Version 11.1**

### **User's Guide**

© 2017. April. All rights reserved. Version 11.1

Loftware, LLM, Loftware Label Design, Loftware Print Server, LPS, Loftware Connector, Global Marking Solutions, I-Push, and I-Pull are all registered trademarks of Loftware, Inc. Loftware WebAccess, LWA, and Loftware Web Services are trademarks of Loftware, Inc. SAP is a registered trademark of SAP AG in Germany and in several other countries. Oracle and Java are registered trademarks of Oracle and/or its affiliates. All other marks are the property of their respective owners.

Loftware, Inc., 249 Corporate Drive, Portsmouth, NH 03801  
Phone: 603.766.3630  
Fax: 603.766.3631  
sales@loftware.com  
www.loftware.com

## Contents

---

<b>The Software Print Server</b> .....	<b>5</b>
Software Label Manager .....	5
Software Documentation .....	6
Contact Software .....	7
Technical Support .....	7
Licensing, Warranty, and Support .....	9
<b>Understanding the LPS Architecture</b> .....	<b>11</b>
The Front End .....	11
The Back End .....	12
The Interface and the Transparency Line .....	13
Software System Requirements .....	15
Software Print Server .....	16
Performance Considerations .....	21
System Analyst Questions .....	24
Cluster Installation Information .....	25
<b>About Configuring the LPS</b> .....	<b>27</b>
To open the Software Print Server Configuration Utility .....	27
General Tab .....	27
Information .....	31
Logging .....	32
Notification and Status Reporting Tabs .....	38
Housekeeping Tab .....	38
<b>Label Versioning</b> .....	<b>43</b>
Considerations .....	43
Promotion .....	43
Enable Label Versioning .....	44
<b>About LPS Modes</b> .....	<b>47</b>
Service Mode .....	47
Interactive Mode .....	48
<b>Print Request Data Structures</b> .....	<b>51</b>
PAS and CSV Commands .....	51
XML Files .....	58
<b>Maximizing System Flexibility, Minimizing Maintenance</b> .....	<b>65</b>

---

---

About Data Push .....	65
<b>About LPS Interfaces .....</b>	<b>71</b>
The File Interface .....	71
ActiveX / .NET Interface .....	72
Loftware Connectors .....	72
Direct Socket Interface .....	73
TCP/IP Socket Interface .....	74
<b>About the Loftware Print Server Client Applications .....</b>	<b>77</b>
Notification Agent .....	77
LPS Status Client .....	77
On-Demand Print Client .....	77
LPSSend Client .....	77
ActiveX Client Control .....	77
Internet ActiveX Control .....	78
Loftware .NET Control .....	78
About the Notification Agent .....	78
On-Demand Print Client .....	89
About the Status Client .....	99
<b>Hints and Troubleshooting .....</b>	<b>114</b>
LPS Service .....	114
Logging .....	114
Scan Directories and File Drop .....	114
Emergency Mode .....	115
<b>Loftware Utilities .....</b>	<b>118</b>
Loftware Version Utility .....	118
LPSSend Interface .....	119
WDPing Diagnostic Utility .....	126
Loftware Reporter Utility .....	131
Backup and Restore Utility .....	138
<b>Internet Printing .....</b>	<b>144</b>
Loftware's WebPush / Web Listener .....	144
Loftware's WebClient .....	144
Internet ActiveX Control .....	144
Internet Printing Considerations .....	144
The WebClient .....	145
Definition of Terms .....	163
Using the WebClient with a Database .....	165
Internet Data Push and the Web Listener .....	167

---

---

<b>The ActiveX Client Control</b> .....	<b>184</b>
Installation and Use of ActiveX Client Control .....	185
ActiveX Client Control Properties .....	188
ActiveX Client Control Methods .....	206
ActiveX Client Control Events .....	212
ActiveX Client Control Properties and Methods .....	218
<b>Internet ActiveX</b> .....	<b>222</b>
Installation/Use of the Internet ActiveX Control .....	223
Design Scenario and Distribution .....	225
Internet ActiveX Properties .....	226
Internet ActiveX Methods .....	236
Internet ActiveX Events .....	241
Internet ActiveX Reference Table .....	246
<b>The .NET Control</b> .....	<b>250</b>
Considerations when using the .NET Control .....	251
Installation and Use of the Software .NET Control .....	251
Distributing the Software .NET Control .....	254
Software .NET Control Methods .....	255
Software .NET Control Properties .....	271
Software .NET Control Events .....	278
Reference Table for the Software .NET Control .....	280
<b>External Links</b> .....	<b>283</b>

The Software Print Server® (LPS) is a scalable high-speed, high-volume marking solution for barcode labels and RFID labels. The LAN/WAN and Internet adaptable technology can act as both a middleware solution for automated business processes and as a back-end component for small, medium, and large-scale business systems that desire per-client printing and encoding.

The Software Print Server's server-centric approach simplifies administration to a centralized location on the network. Configurations, images, labels, layouts, serial files, and printers can be managed on a single computer or as part of a cluster for fail-safe redundancy as described in the *Clustering* section of the *Software Print Server and Label Manager Installation Guide*.

The Software Print Server automates printing labels from front end systems: EDI/ASN, Pick-Pack, ERP, MRP, Wireless, WMS and custom systems, regardless of the platform on which they reside. Host applications running on operating systems, such as UNIX, AS/400, HP-UX, Linux, Solaris, can output reports that can be pushed (FTP), pulled (Polling) or bi-directional (TCP/IP Socket, Software Connector® for Oracle and Software Connector for SAP) to the LPS interface.

## Software Label Manager

Software Label Manager is a suite of software applications for designing and printing barcode labels. Software's stand-alone printing modules, On-Demand and Range Print, can be used with Software Label Manager for non-automated/low-volume barcode label printing. New printer drivers are continually being added; please contact [support@loftware.com](mailto:support@loftware.com) for availability of new drivers for printers and RFID devices.

The Software Label Manager applications are included with all licensing models of the Software Print Server. The LPS includes the tools necessary for high volume / automated printing. With the Software Label Manager, you can:

- Create barcode and RFID labels
- Set up and configure barcode and RFID devices
- Customize labels to meet compliance standards
- Use single and double byte character sets in your labels to support Korean, Japanese and other Asian languages
- Build MaxiCode, GS1-128 (Formerly UCC-128), TLC39, and QR Code barcodes
- Define data for RFID labels and tags using EPC or DoD encoding
- Print labels as needed using the On-Demand print module
- Extract label data from different sources, including ODBC 32 compliant databases
- Use wizards for many of the previously mentioned features

## Software Documentation

### Software Label Manager

The *Software Label Manager User's Guide* describes label creation and printing using Software Label Manager, including:

- Label design
- Barcodes
- Device Connections
- RFID Field Encoding, and Smart Label Printing
- On-Demand Printing
- Templates and Wizards
- Double-byte Character Sets
- Databases and ODBC
- Range Printing

### Software Print Server

The *Software Print Server User's Guide* contains advanced information for those seeking an understanding of Software's enterprise solutions. This guide is designed to give those who are already knowledgeable about label printing a head start in implementing a printing solution using advanced technologies. Included in this guide is information about:

- The Software Print Server
- Thin Clients
- Internet Printing
- LPS Clustering and redundant systems
- Client, Internet, and .NET Control

The Software applications listed previously may be run in English, French, German, or Spanish.

**Note:** You must purchase and license the Software Print Server separately from the Software Label Manager.

### Software.com

Visit [www.software.com](http://www.software.com) for the latest revisions of the Software Print Server and Software Label Manager user guides. Also, visit the [Software Print Server Family Knowledge Base](http://help.software.com) at [help.software.com](http://help.software.com) for additional information and tips on a variety of subjects.

## Contact Software

Loftware, Inc.  
249 Corporate Drive  
Portsmouth NH 03801  
U.S.A.

### Professional Services

For consultation, implementation services, training or product optimization please contact Loftware's Professional Services Group.

**Phone** +1.603.766.3630 x209

**E-mail** psg@loftware.com

### Technical Support

For installation and configuration questions, please contact Loftware's Technical Support department. Visit [www.loftware.com](http://www.loftware.com) for Loftware's technical support policies.

**Phone** +1.603.766.3630 x402

**Fax** +1.603.766.3635

**E-mail** support@loftware.com

### Customer Service

For licensing, product information, and ordering questions, please contact Loftware's Customer Service department. Please have your Serial Number and Registration information available, so we can provide service to you quickly and efficiently.

**Phone** +1.603.766.3630 x401

**Fax** +1.603.766.3631

**E-mail** customerservice@loftware.com

## Technical Support

Software licenses purchased directly from Loftware include the first year of Technical Support. This initial 12-month support period starts on the day the product is shipped and invoiced from Loftware's factory. When needed, support recipients during this period are eligible to receive unlimited telephone support, access to software upgrades and enhancements and speak with our Systems Analysts.

### Premium Annual Support Contract

To ensure uninterrupted telephone support as well as access to the latest software upgrades and enhancements, make sure all your software licenses remain under a Loftware Support Contract. After your first year of ownership, you will be sent a notice to renew your support contract. Please refer to Loftware's Web site for additional information about this very important topic, or if you prefer, call

Loftware's Customer Service Department for more information.

During the one-year Support Contract period, Contract Subscribers have access to the following services:

1. Unlimited Technical Support Incidents
2. Access to Loftware's Professional Services Group
3. Automatically eligible to download software upgrades and service packs from our Web site
4. Automatic e-mail notification when new versions of software become available
5. When necessary, access to senior Loftware technical support staff, via phone and e-mail
6. Guaranteed software license replacement for accidentally damaged or malfunctioning hardware keys

## Before Calling Support

Loftware has highly trained technicians available to help you with your labeling system. Technical support calls are not accepted until all of the following Technical Support requirements are met:

1. Your product is registered. If you have not registered your software, you may do so at <http://loftware.com> or via fax by using the form included with your software.
2. There is a Support Contract in place that covers the specific license in question.
3. You have checked the user's guide(s) for your answer. If you do not have the User's Guides, both of the guides can be downloaded in PDF format from our web site or read on-line. User manuals are also on the Loftware CD.
4. You have checked the Loftware's Knowledge Base articles on our <http://loftware.com>. Hundreds of frequently asked questions and typical problems are documented there in easy to read articles.
5. If you suspect that your problem is hardware related, try to first determine if it is a problem with your computer, Network, or printer and contact the appropriate company. Loftware does not sell or service any hardware products.
6. Have your serial number and version number of the product you are using ready. These numbers can be obtained by accessing the Help|About menu of the label design mode.
7. Think about how you are going to efficiently explain the problem prior to speaking with a technician. The better the description, the quicker the solution and/or resolution to your problem.
8. If this is a follow up call to a previous incident, please have the incident number ready.

**Phone** 603-766-3630 Choose Option 3

**Fax** 603-766-3631

**E-mail** [support@loftware.com](mailto:support@loftware.com)



## **Licensing, Warranty, and Support**

The following documents are available on the CD-ROM or Internet download of the Software:

- Software End User License Agreement
- Software Third Party Terms and Conditions
- Software Services Support Agreement

This page intentionally left blank

---

## Understanding the LPS Architecture

---

### The Front End

Think of your existing host application as the Front End of your labeling system. Your host application may be a WMS/ERP application or a homegrown application. The following figure is one possible front-end network layout. Your network may be different.

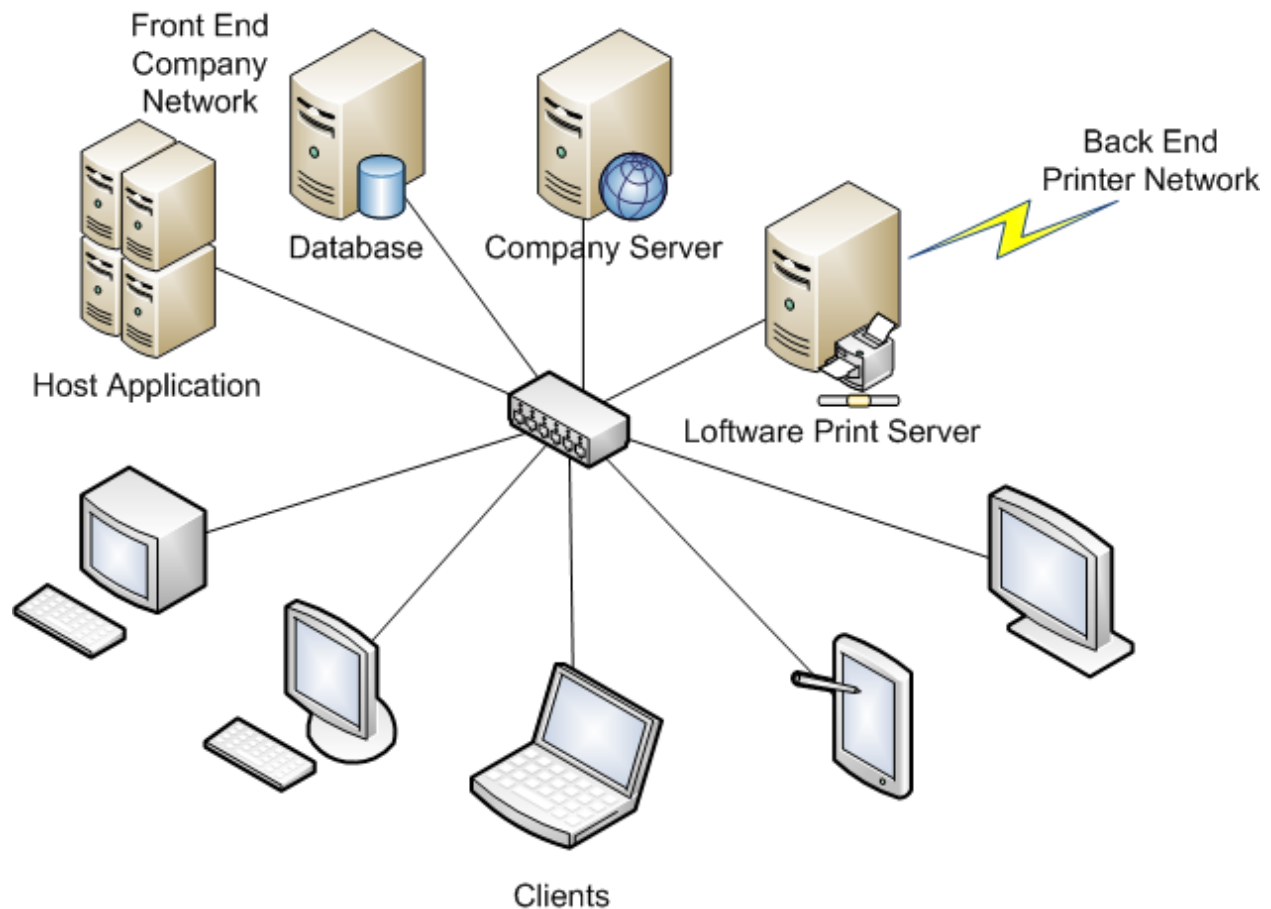


Figure 2.1: A typical company network with link to a Software back-end printer network.

The Front End application decides the following:

- The label format (template) to print
- The data for the format
- The number of labels to print
- The destination printer

When these decisions are made, the label-printing event is triggered by sending a request from your application to the LPS via one of its interfaces. As explained in the next section, the interface ties the front and back ends of your system together.

## The Back End

Think of your devices, device connections, and LPS middleware installation as the Back End of your marking system. Your label designs and associated files are included in this definition of Back End. The following figure is one possible back-end network layout. Your network may be different.

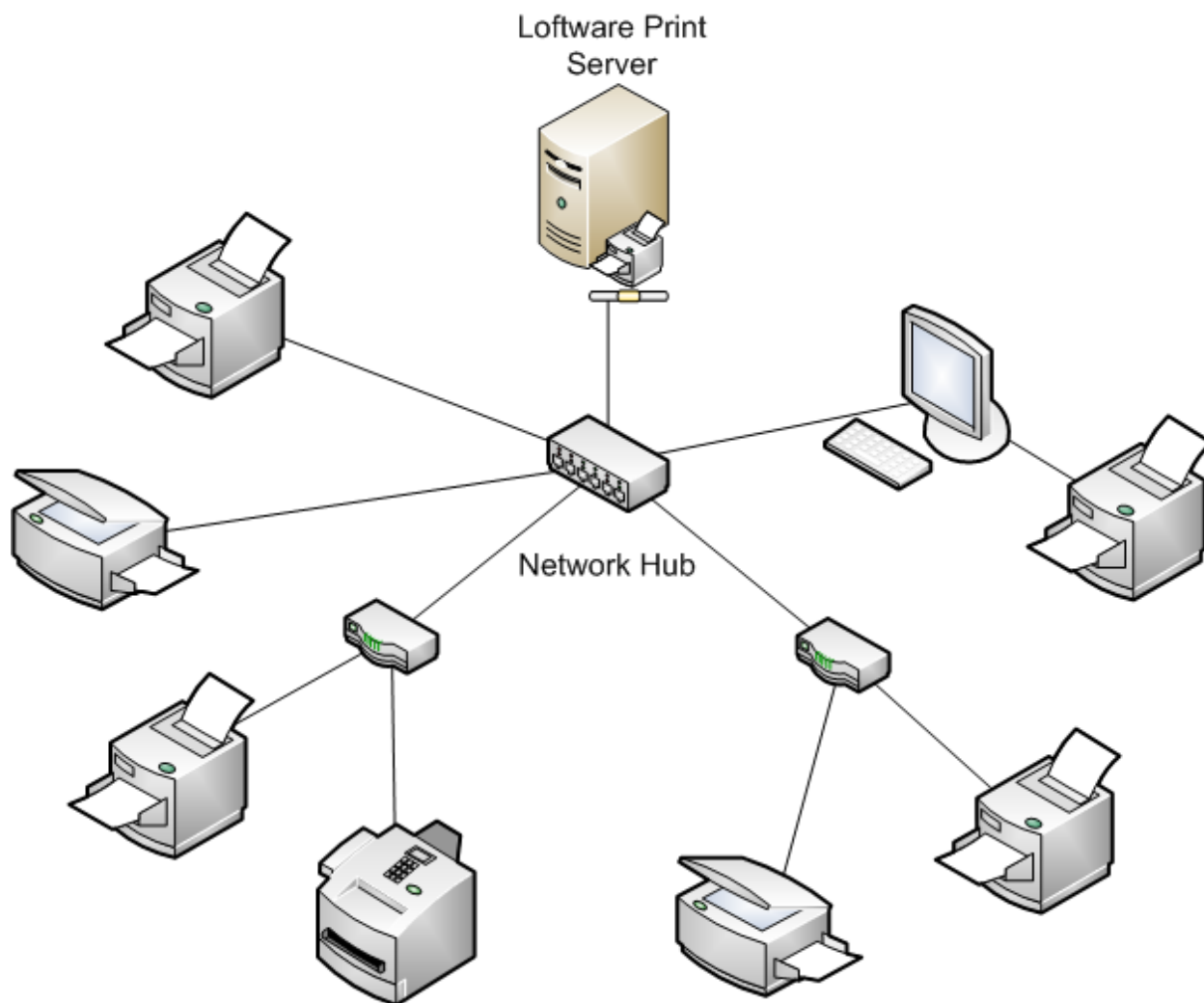


Figure 2.2: Back End of an Enterprise-Wide Barcode Labeling System.

This diagram shows several banks of devices connected to the network, each with its own TCP/IP network address. The printers could be attached to shared workstations instead of using TCP/IP if the printer is going to be shared with other applications. With the correct Loftware license key, an unlimited number of devices can be addressed by a single LPS server.

# The Interface and the Transparency Line

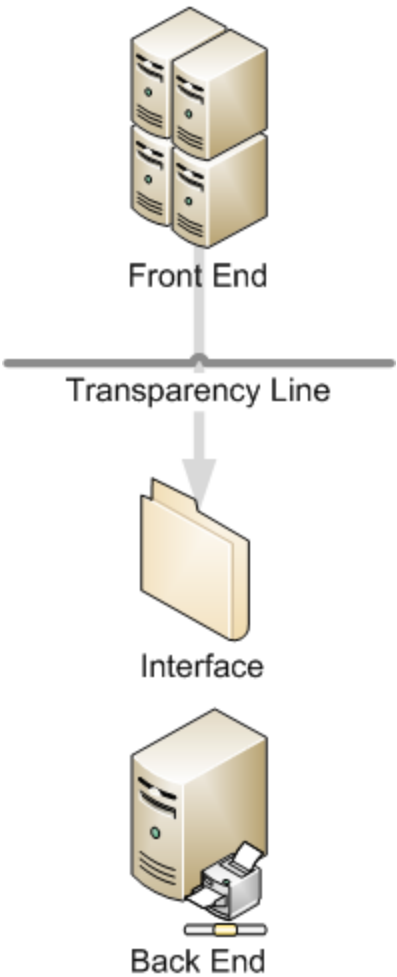


Figure 2.3: The relationship between the Front End and the Back End and the Interface

## The Interface

The interface is what allows your Front End program to communicate with the LPS. The Front End makes decisions on what labels to print based on various criteria. Once this information is known, it is communicated through the interface to the Back End where it is processed. There are several different types of interfaces to choose from, each having its own benefits.

Interface	Benefits
File	The File interface allows Front End applications to make requests to the LPS via a file drop to a shared network drive. The LPS detects the file and processes the request. Any program in any language can use this interface.

Interface	Benefits
TCP/IP Socket	The TCP/IP Socket interface allows Front End applications to have bi-directional communications directly with the LPS through a socket without the need for file transfer or shared drives. Many aspects of the LPS configuration are available to the front end program through the socket, such as Label List, Field List, Device Configurations, etc. Device status and EPC data pass back are also available in this interface. This interface is for advanced programmers only and is only available in the LPS Premier product. This interface is usually used by UNIX programmers but can be used by C++ or C# programmers as well.
Direct Socket	The Direct Socket interface is similar to the TCP/IP interface mentioned previously. The difference is that it is much easier to code to because it is unidirectional. Print Requests are assembled in an XML file and 'shot' through the socket. No status is available other than the fact that if you cannot open the socket, the LPS is probably not running. This interface is usually used by UNIX programmers, but can be used by C++ or C# programmers as well.
ActiveX	The ActiveX Interface and the .NET Interface allow programming languages to send printing requests to the LPS from anywhere on the LAN, WAN, or Internet. Under the covers, these tools actually use a socket connection for speed and reliability. These tools are available to Windows applications.
On Demand Print Thin Client and the Web Client	The On Demand Print Client and the Web Client allow label requests to be made from any client computer on the network or across the Internet by prompting the operator for keyboard or database key information. No programming is required to use these client programs. See Chapters 2 and 3 for detailed information on these.

## The Transparency Line

The transparency line depicted in Figure 1-C indicates that the Front and Back ends are transparent to one another. Applications do not need to know anything about the printers or how they are connected. Likewise, the Back End has no knowledge of what comprises the Front End or how it operates.

The transparency line is a critical factor for making the LPS effective as a middleware solution. With this, Software technicians, analysts, and engineers do not need to have detailed knowledge of your Front End system in order to support you.

By following the procedures outlined in the following sections, you can achieve remarkable flexibility in your system without having to cross the line. The LPS takes care of the printing details so that Software customers can concentrate on their business processes.

### Related Information

For detailed information on the LPS interfaces, refer to the *LPS Interfaces* section in this guide.

## Loftware System Requirements

System requirements are dependent on a number of factors, including:

- Concurrent users
- Printers driven
- Volume of labels printed

As load increases, memory, processor speed and the number of processors may need to be adjusted. Loftware's Professional Services Group (PSG) can help determine your system requirements.

### Virtual Environment Support

Loftware provides limited support for products running on operating systems (compatible with Loftware products) in a virtual environment, running VMware®. For VMware specific support or issues, please contact VMware support directly or via your vendor.

Loftware requires that companies seeking to use any of the Loftware software products in a virtual operating system environment purchase a license for each Loftware product used in every virtual environment in accordance with the license model for that Loftware product. That is, an installation of a Loftware product running within a virtual operating system environment must be licensed to the same terms as a Loftware product running in a physical (non-virtualized) one.

Loftware products running in a virtual environment require software license keys and a static MAC address for each virtual operating system environment.

### End of Support Notices

Please review the [End of Support Notices](#) on [loftware.com](#). If you are running Loftware solutions on one of the systems listed in the End of Support Notice, you are encouraged to upgrade to a supported system.

## Loftware Print Server

The following are the system requirements for Loftware Print Server® 11.1.

### Server Requirements

Consider these requirements the minimum to run the Loftware Print Server (LPS). Medium and high-volume systems may require more or faster processors and higher available RAM and disk space in order to achieve desired performance and throughput. LPS will perform best when not limited by available RAM, disk space, or processor speed.

To optimize server utilization when using Print Groups, consideration should be given to the server platform size, configuration and available resources. Loftware recommends having a server configuration of:

- **Processor Cores:** Two cores per Printer Group plus one core for the system
- **RAM:** 2GB per Printer Group plus 4GB or the system

**Important:** LPS is not supported on any server that is acting as a Domain Controller.

Component	Requirement
Computer Processor Speed	2.0 GHz Quad Core
Computer Memory (RAM)	4GB, 8GB Recommended
Computer Hard Disk Space	Minimum 4GB Available Disk Space
Networking	IPv4 (IPv6 not supported)
Operating System	<b>Microsoft Server Operating Systems</b> Windows Server 2016 (nonclustered) Windows Server 2012 R2 Windows Server 2012 Windows Server 2008 R2 SP 1 (64-bit) Windows Server 2008 SP 1 or later (32-bit or 64-bit) <b>Note:</b> LPS and LLM products, controls, clients, and connectors run as 32-bit applications on 64-bit Windows operating systems.
Connected Devices	License dependent
Virtualization	VMWare support



## Recommended System Specifications

The following specifications apply to LPS environments with a single printer group of less than 500 printers. If your environment requires multiple groups and more than 500 printers, please contact Loftware for assistance determining the appropriate system requirements for your configuration.

Operating System	Low Volume	Medium Volume	High Volume
Windows Server 2016 (nonclustered)	2.0 GHz Quad Core 8 GB RAM 20 GB Available Disk Space	2.0 GHz Quad Core 12 GB RAM 50 GB Available Disk Space	2.0 GHz Quad Core 20 GB RAM 100+ GB Available Disk Space
Windows Server 2012 R2 and Windows Server 2012	2.0 GHz Quad Core 4 GB RAM 20 GB Available Disk Space	2.0 GHz Quad Core 8 GB RAM 50 GB Available Disk Space	2.0 GHz Quad Core 16 GB RAM 100+ GB Available Disk Space
Windows Server 2008 R2 and Windows Server 2008	2.0 GHz Quad Core 4 GB RAM 20 GB Available Disk Space	2.0 GHz Quad Core 8 GB RAM 50 GB Available Disk Space	2.0 GHz Quad Core 16 GB RAM 100+ GB Available Disk Space

## Client Requirements

The Loftware Print Server Clients include:

- On Demand Print Client
- Loftware Print Server Status Client
- ActiveX

**Important:** The version of the LPS client being used must match the version of the LPS the client is being used with. For example, only version 11.1 clients are supported with version 11.1 of the LPS.

## Loftware Print Server Client Requirements

Component	Requirement
Computer Processor Speed	2.0 GHz Dual Core
Computer Memory (RAM)	2 GB
Computer Hard Disk Space	Minimum 2GB Available Disk Space

Component	Requirement
Operating System	<p><b>Microsoft Client Computer Operating Systems</b></p> <p>Windows 10</p> <p><b>Note:</b> Mobile devices running Windows 10 are not supported.</p> <p>Windows 8.1</p> <p>Windows 7 SP 1 (32-bit)</p> <p>Windows Vista SP 2 (32-bit)</p> <p><b>Microsoft Server Operating Systems</b></p> <p>Windows Server 2016 (nonclustered)</p> <p>Windows Server 2012</p> <p>Windows Server 2008 R2 SP 1 (64-bit)</p> <p>Windows Server 2008 SP 1 (32-bit or 64-bit)</p> <p><b>Note:</b> LPS and LLM products, controls, clients, and connectors run as 32-bit applications on 64-bit Windows operating systems.</p>
Notes	The ActiveX client control can only be used by 32-bit applications.

## .NET Control Requirements

Component	Requirement
Computer Processor Speed	2.0 GHz Dual Core
Computer Memory (RAM)	2 GB
Computer Hard Disk Space	Minimum 2GB Available Disk Space

Component	Requirement
Operating System	<p><b>Microsoft Client Computer Operating Systems</b></p> <p>Windows 10</p> <p><b>Note:</b> Mobile devices running Windows 10 are not supported.</p> <p>Windows 8.1</p> <p>Windows 7 SP 1 (32-bit)</p> <p>Windows Vista SP 2 (32-bit)</p> <p><b>Microsoft Server Operating Systems</b></p> <p>Windows Server 2016 (nonclustered)</p> <p>Windows Server 2012 R2</p> <p>Windows Server 2012</p> <p>Windows Server 2008 R2 SP 1 (64-bit)</p> <p>Windows Server 2008 SP 1 (32-bit or 64-bit)</p> <p><b>Note:</b> LPS and LLM products, controls, clients, and connectors run as 32-bit applications on 64-bit Windows operating systems.</p>
Operating System Features	Microsoft .NET 4.5 Framework

## Notification Service Requirements

Component	Requirement
Computer Processor Speed	2.0 GHz Dual Core
Computer Memory (RAM)	2 GB
Computer Hard Disk Space	Minimum 2GB Available Disk Space

Component	Requirement
Operating System	<p><b>Microsoft Operating Systems</b></p> <p>Windows 10</p> <p><b>Note:</b> Mobile devices running Windows 10 are not supported.</p> <p>Windows 8.1</p> <p>Windows 7 SP 1 (32-bit)</p> <p>Windows Vista SP 2 (32-bit)</p> <p><b>Microsoft Server Operating Systems</b></p> <p>Windows Server 2016 (nonclustered)</p> <p>Windows Server 2012 R2</p> <p>Windows Server 2012</p> <p>Windows Server 2008 R2 SP 1 (64-bit)</p> <p>Windows Server 2008 SP 1 (32-bit or 64-bit)</p> <p><b>Note:</b> LPS and LLM products, controls, clients, and connectors run as 32-bit applications on 64-bit Windows operating systems.</p>

## Performance Considerations

The Software Print Server's (LPS) scalable architecture allows it to keep many printers busy at the same time. Because of the process involved in printing across local and remote enterprises, many factors affect LPS performance. This section provides guidelines for what you can expect using your particular system configuration.

The LPS has been extensively tested with many different scenarios, none of which may exactly match your environment. Therefore, the information in this section is of a general nature only. The actual results in your situation will vary.

### Hardware Factors

#### Number of programs your server is actively running

The LPS is performing many tasks when it translates your printer independent requests into the native printer languages necessary to print them. The speed at which your label requests are serviced is proportional to the processor clock cycles allocated (by Windows) to the LPS. The more processor time it gets, the faster it is. In other words, if you are running other server type applications like a database or email server, delays can be experienced when the other applications are busy.

#### Number of processors

The Software Print Server is written to take advantage of multiple core processors on a computer running Windows Server 2008 or higher. Because it is a multi-threaded application, the operating system balances the threads among the available processors. The net result is that the more processors you have, the more throughput performance you should be able to achieve up to the point where the printers' performance becomes the limiting factor.

#### Available Memory

Memory is an important factor in printing speed. Do not attempt to run the LPS system on a server with less than 2 GB. The best way to gauge memory is to use the performance monitor in the Windows Task Manager. Be advised that when the LPS first starts, it is at a baseline memory footprint. The more devices and the larger the print queues get, the more memory will be used. For more information, see ["Recommended System Specifications" on page 17](#).

#### Processor Speed

Printing throughput is proportional to processor speed. A faster processor is a better processor. Do not attempt to run the LPS on processors less than 2.0 GHz Dual Core.

#### Network Speed

If you are dropping requests to a scanned network drive and/or printing to printers on your network, network speed and traffic are going to factor in. Remember, the faster the network, the greater the speed. Making printer connections across a WAN may prove to be slower than LAN connections depending upon traffic, data, and other factors. This can be avoided by dedicating an LPS server to each

LAN, although your file drops may still be coming over the WAN. Real world experience has shown us that WAN speed is entirely acceptable for most applications.

## **System Failover Protection**

Companies who require their printing operations to be up 7x24x365 should consider the economic impact to short and longer-term printing outages. If this is unacceptable to your organization, Loftware encourages you to consider making additional investments in software licensing to deploy a system failover strategy.

A well thought out failover strategy along with a disaster recovery plan (DR) can be best attained by configuring two identical LPS servers, one for primary use and one designed for an active failover. In all cases when configuring additional servers, an equal number of printing devices will also need to be added.

There are different technical approaches to achieving system redundancy in your printing environment. Because of this, we recommend that you contact Loftware directly to arrange a time to speak with one of our Loftware Professional Services Analysts. The analyst will be able to assist you and make the proper recommendations.

## **Devices and Device Connections**

### **Number of physical devices you are driving**

LPS has the potential to service an unlimited number of configured devices, depending on the license. The LPS's scalable architecture allows you to keep all the devices working at the same time, but a performance penalty proportional to the number of devices you are driving is paid.

Print groups enables the grouping of print devices when managing a large number of print devices on a single LPS server. In addition to improving the management of large printer deployments, large print jobs that may have been memory-constrained will see improved performance by taking advantage of the new multi-threaded printer group architecture that better utilizes server processing and memory.

### **Native drivers versus Windows drivers**

Using the native Loftware printer drivers for supported printers is always faster than using Windows printers. The reason for this is that we use the printers' high speed, native language as opposed to sending a bitmap for Windows printers.

### **Brand names of the printers**

Loftware maintains a position of hardware neutrality when it comes to thermal transfer printers. We do not sell or recommend printers. We do say that some printers are much faster than others are when it comes to imaging a request. Do not base your decision on the documented print speed (inches per second) of your printers alone. This specification is for printing multiple copies of the exact same label without having to image new data. When purchasing printers, consider imaging (processing) speed, as well as inches per second.

## Labeling Factors

### Number of Fixed Fields on the Label

The LPS pre-downloads fixed fields (lines, boxes, fixed images, and fixed text) from your label to the printer. Subsequent requests for the same label only download the variable data that has changed. Please note that some thermal transfer printers and all Windows printers do not support this capability. The number of fixed fields on your label affects the first download of the label but not subsequent prints. The number of variable fields and whether or not you use logos or graphics has a much larger impact.

### Number of Variable Fields on the Label

Variable data fields on the label are sent to the printer every time one of them changes. If you are printing multiple copies of the same label, expect very fast throughput. If your variable data changes between labels, two factors influence print speed. First, the new data fields must be sent to the printer. Secondly, the printer must take the time to re-image the label with the new data before it prints.

### Using Fixed Logos or Graphics

An example of a fixed graphic might be a company logo on the label that is static for all labels. Most thermal transfer printers support the pre-downloading of logos so there is a download penalty for the first download, but once the graphic is there, it is not re-downloaded. Graphics are data intensive and take longer to download than normal barcode, text, and line/box fields.

### Using Variable Graphics or TrueType Fonts

If your label contains variable text fields that are formatted with TrueType fonts or graphics that change with each label, throughput may be significantly reduced. Although the LPS system can handle labels like this, the time it takes to download new labels makes this prohibitively slow. Software strongly recommends that you use the printers' native fonts, some of which are smooth, like TrueTypes, but are native to the printer.

### Complex Formulas and Serial Numbers

If you have fields on your label that are incrementing, or have a data source defined as formula, you may pay a throughput penalty. If the printer is capable of performing the incrementing, the job is passed to it and no penalty is incurred. If, however, the printer is not capable of incrementing, then the LPS must increment the data and re-send the variable fields between labels thus delaying throughput.

## System Analyst Questions

If you have questions regarding setting up an enterprise printing system, call the Loftware Professional Services Group at (603) 766-3630. The Analyst reviews your requirements and addresses any concerns that you have. Analyzing your needs up front ensures that your printing system is designed properly the first time. Before you call, please answer the following questions.

**Note:** Emailing copies of your label(s) is also very helpful.

Question	Answer
How many printers do you intend on driving?	
What kind of printers do you have?	
How many Printer Groups will you use?	
Do you use any printers with Windows Drivers?	
Do your labels have any fixed graphics on them such as a company logo?	
Do your labels have variable graphics on them such as a picture of a part?	
Approximately how many different label formats do you have?	
Approximately how many fixed and variable text and barcodes are on each label format?	
Do any of your labels have incrementing data fields, such as Serial Numbers?	
Are all of your printers in the same building?	
How often do you print multiple copies of identical labels to one printer in one request?	
How often are you requesting numerous different labels to one printer in one request?	
What computer configuration do you plan to run the LPS on? (GHz, memory, processor)	
What Operating System is the server running?	
Is the computer dedicated to label printing?	

### Related Information

For more information on system planning, refer to the *Performance Considerations* section of this guide.



## Cluster Installation Information

For information on installing the Software Print Server on a Windows Server 2008 or Windows Server 2012 cluster, refer to the *Software Print Server and Software Label Manager Installation Guide* located on the installation CD, download package, or [loftware.com](http://loftware.com).

This page intentionally left blank

You can start, stop, and configure the LPS using the Software Print Server Configuration Utility.

### To open the Software Print Server Configuration Utility

- Open LPS Configuration from the **Start** menu under Software Labeling

### General Tab

The purpose of the General tab is to set up additional scan directories for programs that cannot write to the default wddrop directory. Before Scan directories can be configured or added, LPS must be stopped. LPS defaults to scanning a directory called WDDrop, which is a directory under the primary Software directory. This default directory is necessary for On Demand Print and ActiveX Client printing and should not be changed. You may, however, add scan directories that are local or on the network.\*

**Note:** LPS runs in memory as a service; therefore, most mapped drive letters, due to new stringent security standards, will not be accessible. When configuring additional scan directories, instead of using a mapped drive (M:\drop), use the Universal Naming Convention (UNC) path (\\ws1\drop).

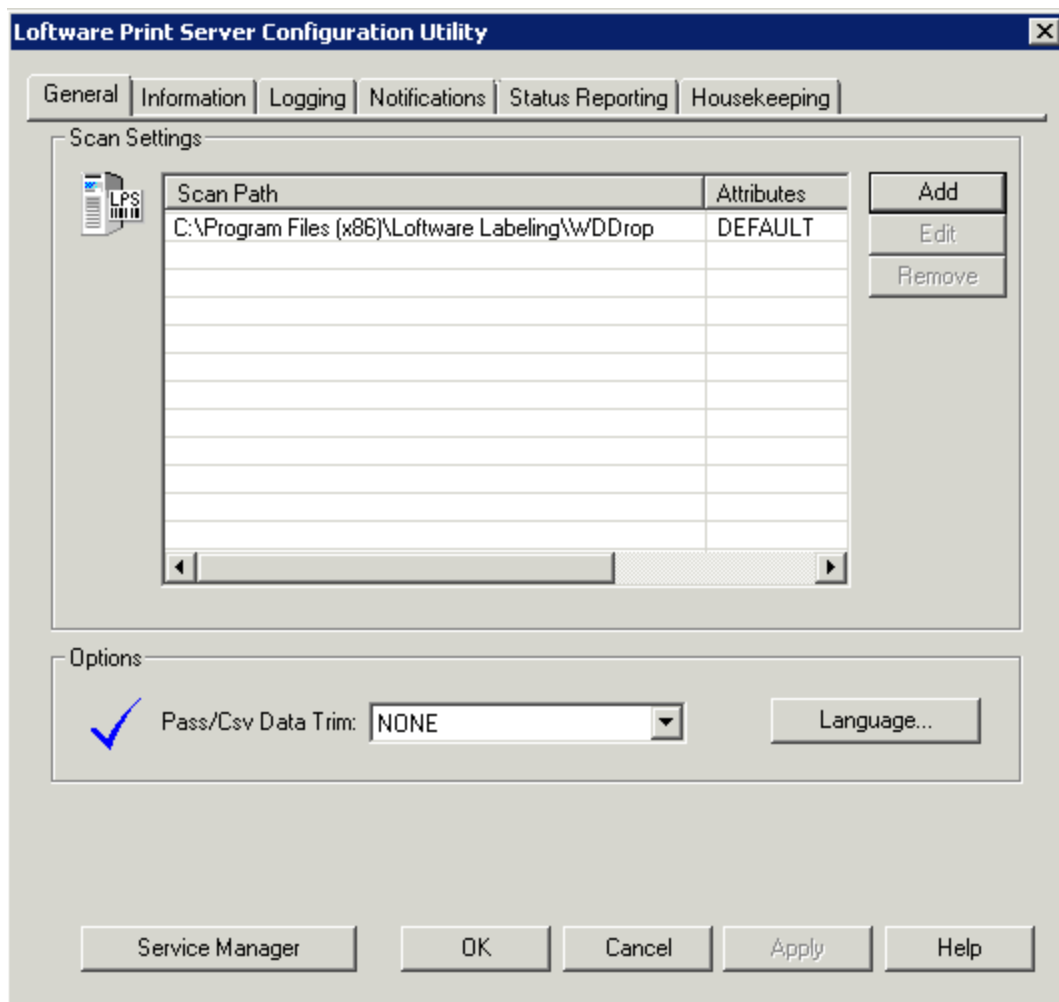


Figure 3.1: The LPS Configuration Utility showing the General tab

## Scan Settings Section

### Multi-Scan Grid

This is for the File Interface only. Consider the following scenario: There are 50 printers controlled with the Software Print Server. Requests for labels happen in on demand. Let us say that your UNIX server has made 500 different label requests for 40 of the printers. At the same time, an on-demand operator wants to print one label on printer 41. If you were scanning only one directory, there may be a considerable delay before the single request is processed, because of the hundreds of requests that are in queue before it.

There are several ways of setting up scan directories for optimizing system performance. One way is to have a separate scan directory for every printer, and drop file requests to the appropriate directory depending on the target printer for which the request is made. All scan directories are serviced simultaneously, so that in the previous scenario, each single request buried under multiple large requests is serviced quickly and with no noticeable delays. This is referred to as multi-threaded input and although easy to set up, it represents an advanced feature found only in the Software Print Server.

The concept of 'multi-scan' is not an exact science. The best advice we can give is to carefully consider all aspects of your labeling system and create scan directories based on a logical division of tasks.

**Note:** Simple systems that do not have high throughput requirements can simply use the default of one scan directory.

## Adding a Scan Directory

Before adding a scan directory to the LPS, you must first create it using Windows Explorer.

- Click the **Add** button in the LPS Configuration Utility (General tab). The following dialog box is displayed:

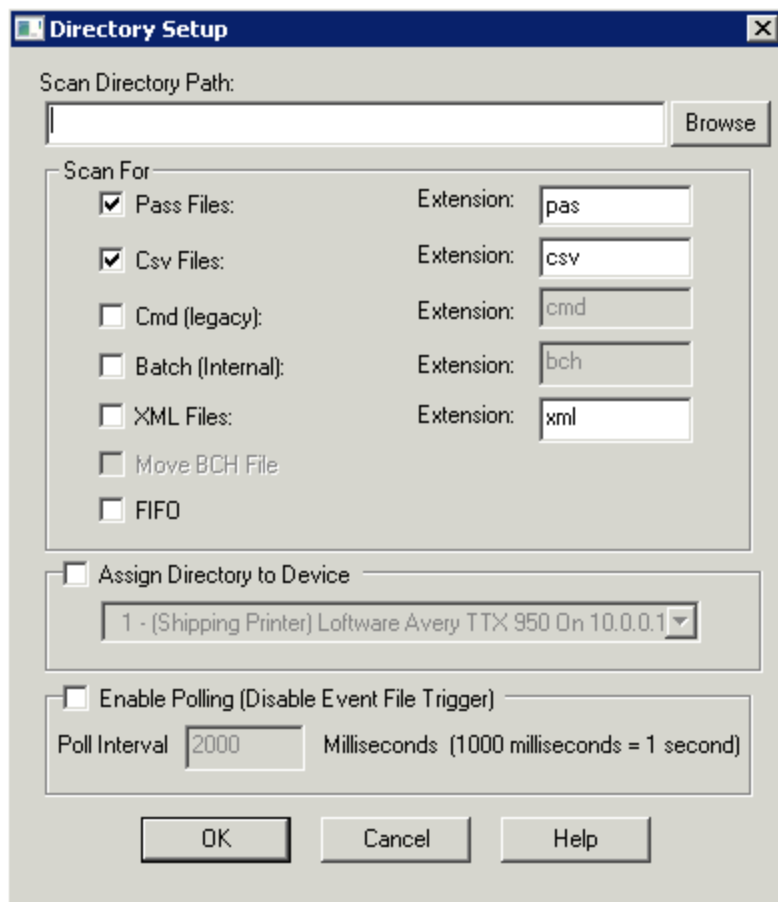


Figure 3.2: Directory Setup Dialog Box

## Scan Directory Path

This is the directory into which print requests are dropped using the File Interface. The directory can reside on any network drive to which both your application and the LPS have Read/Write access. If you are using the LPS in Service mode without logging on, UNC paths like \\remote\server\dir1 are required, but only if the scan directory is on another computer. If the scan directory is local, a UNC path is not required.

Click **Browse** to locate the scan directory. Using the **Browse** button eliminates possible errors when typing the scan directory path manually. The directory must exist before you can actually browse to it. It is not created for you. You may create scan directories using file Explorer.

**WARNING:** Do not write data to a file that is in the WDdrop or other scan directory. Data should be written into a file in a non-scan directory and then moved to the WDdrop or other scan directory. Adding data to a file in a scan directory may result in a job being stranded in the Pending (OLEBP) directory without a corresponding Loftware Print Job (.lpj) file.

## Scan For Section

Option	Description
PAS Files	The LPS scans for PAS files. Fill the extension with the extension for which you wish to scan, that is *.pas. .
CSV Files	The LPS scans for .CSV (default extension) files. Although harder to read, CSV files are much more compact than .pas or .xml files.
Cmd (legacy)	Scan for Command Files and Batch Files - Legacy settings, DO NOT use!
Batch (Internal)	The LPS scans for batch files. These files contain the name of the label with its data, quantity and duplicates.
XML Files	The LPS scans for .xml (default extension) files. This scans for files that have a header row and subsequent data lines.
FIFO	First In First Out - when checked, files are processed in the order in which they are received, and not the order in which they are entirely written to the sectors on the hard disk. FIFO is not guaranteed unless this setting is turned on. FIFO may reduce printing performance.

## Assign Directory to Device Section

This assigns a designated printer to the added directory. This is helpful if it is difficult to include a printer number command in your request. Any files dropped into a directory with this setting turned on are dispatched to the assigned printer regardless of any reference in the file itself. Extra care must be taken to ensure that the file you are requesting is indeed designed for the assigned device.

## Enable Polling Section

Polling must always be enabled for shared network drives. The LPS remains idle until the operating system notifies it that the contents of the scanned directory have changed. If you have polling disabled and find that the LPS is not processing your files, enable polling and set the polling interval to the

millisecond interval that you wish to poll the directory; 1000 milliseconds (1 second) is the default.

## Options Section

### PAS/CSV Data Trim

Valid for PAS Mode only. If your data contains leading or trailing spaces, you can use this setting to trim them. Choices in the drop-down list are:

- None – No leading or trailing spaces are trimmed. This is the default.
- Trailing – Trims trailing spaces only
- Leading – Trims leading spaces only
- Both – Trims both trailing and leading spaces

When would you set it to Both?

#### Example

You have a label that is connected to a database, and the key field on a label has either a leading or trailing space(s) in it. In this case, the database data fields connected to that key field are not found because the program is searching for the key field without the spaces. In this case, selecting Both ensures that no spaces are included in the field.

## Language

Selection of the default language for Software applications (Design, On-Demand, Range, etc.) can be made on a per-user basis. However, when a service such as the LPS is being used, the language change is for the service, not for the user; therefore it becomes the default language for all users.

#### Example

A user logged in to a computer whose default language has been set to English would like the language of the LPS Service to be displayed in their native language of French.

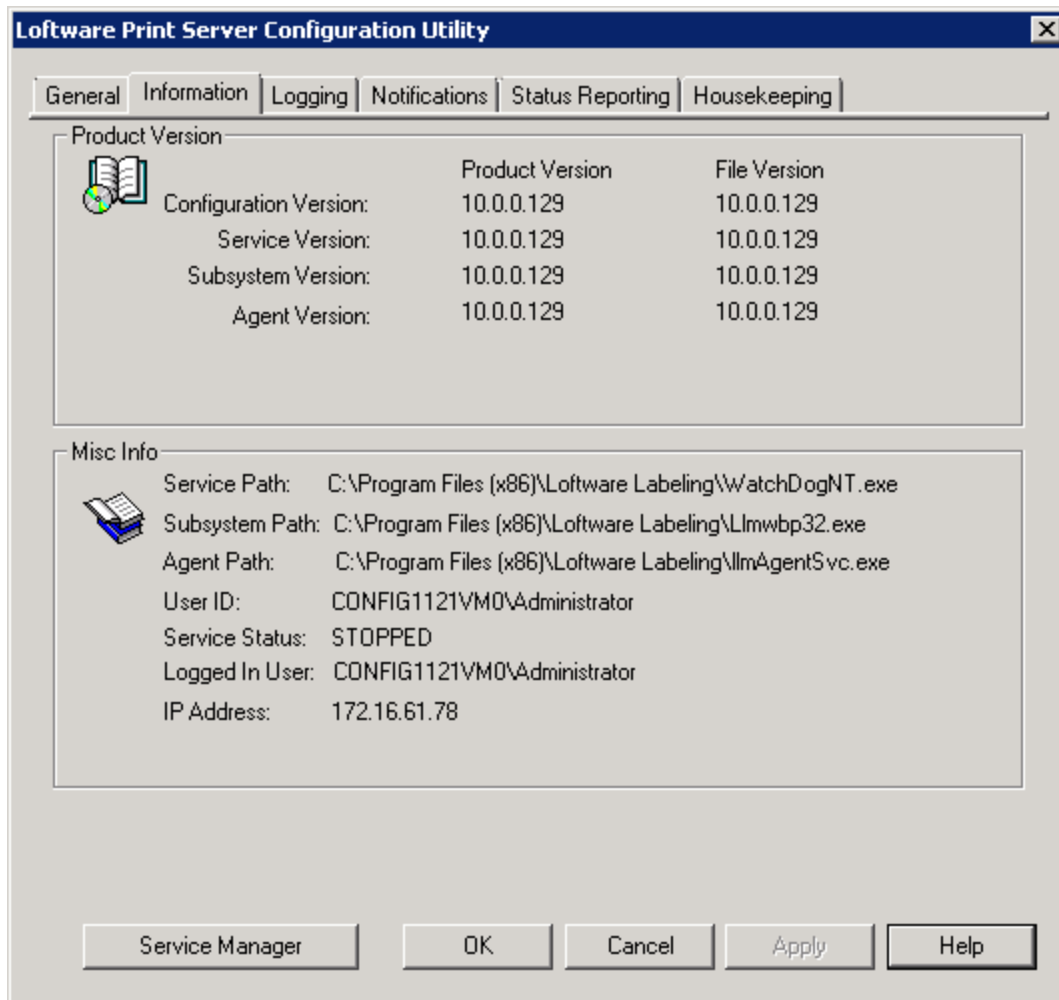
To set the default language for the service to French, the following steps are performed:

1. Open the **Software Print Server Configuration Utility** (LPS Configuration) from the **Start** menu under Software Labeling.
2. Click **Language** under the **General** tab.
3. Select **Français** from the drop-down list.

Remember, this changes the default language for ALL Software users on this computer to French and also changes the language for ALL the other Software services (Web Client, Notification Agent) on this computer as well. This setting controls the user interface only and does not affect LPS performance.

## Information

The Information tab displays product versions and some basic information on your Software Print Server installation.



## Logging

The logging features of the LPS can be used for debugging purposes. It is important to note that, when run as a service, the LPS does not have access to on-screen dialog boxes for displaying error messages. Once your system is in production, this is not an issue. However, during the development of your system, it is important that error and warning messages can be read. Logging assists you in doing this. The following are the types of logging available:

- Event Logging
- LPS Watchdog Logging
- Performance Data Logging

**Note:** System performance can be affected by the amount of information that you log and the frequency that you log the information. Generally, as the quantity and frequency of logging increase, the Software Print Server performance decreases.



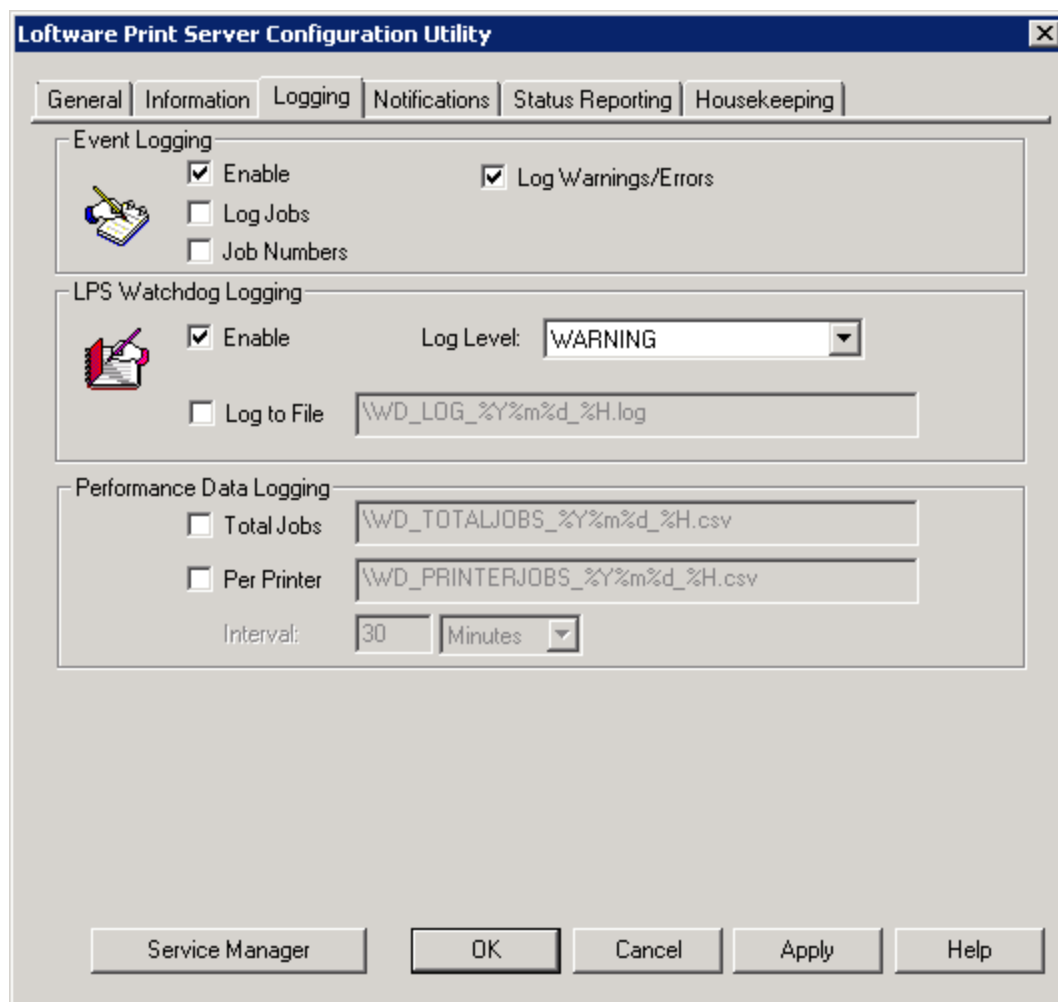


Figure 3.3: Event Logging Tab

## Viewing Log Files

Log files are stored in the Software Labeling Logs folder which varies by operating system. You can access your log files from the **Start** menu under Software Labeling.

## Event Logging Section

Event logging is useful for your own monitoring and debugging efforts. This is helpful when running the LPS as a service because a service does not display messages on the desktop.

The log can be accessed through the Windows Event Viewer. Items are logged to the event log whether you are running in service or interactive mode.

Option	Description
Enable	Enables the logging of informational events to the event log. The exact type of information written to the log is determined by the log command settings described below. Remember, error and warning messages are always written to the log regardless of whether or not this setting is on. This setting also allows access to information about who last changed the configuration of the LPS and when this change was made.
Log Jobs	The log Commands, CSV, and PAS settings write an entry to the event log each time one of these files is successfully processed. Once your labeling system is operational, it is recommended that these settings be turned off. A quick way to do this is to simply clear the <b>Enable</b> setting. The other settings are remembered for the next time you decide to enable logging.
Job Numbers	The printing sub-system returns a unique job number back to the LPS for every submitted job. Turn this option on if you wish these numbers to be recorded in the log. <b>Note:</b> Make sure that the maximum size limit of the Event Viewer is set appropriately, so that the log does not continue to grow until all disk space is consumed. You can set the maximum size limit in the Event Viewer itself.
Log Warnings/Errors	All warning and error messages are written to the event log.

## LPS Watchdog Logging Section

Interactive Logging is only applicable when the LPS has been started in Interactive mode from the **Start** menu. The selections made here are displayed real time in the status window of the LPS. If this setting is disabled, a minimal number of top-level messages are still displayed. We recommended that **On Screen Logging** be enabled with the appropriate settings described below. Only enable the settings that you are interested in. This reduces the amount information you must sort through.

Option	Description
Enable	Enables the logging of informational events to the Software Print Server Interactive window
Log to file	Logs the information displayed in the Software Print Server Interactive window to a file. See the following Formatting and Locating Log Files section.
Log Level	Sets the type of log messages to include. <ul style="list-style-type: none"> <li>■ DEBUG</li> <li>■ INFO</li> <li>■ WARNING</li> <li>■ ERROR</li> </ul>

## Log Levels

Log Level	Information Logged
DEBUG	Includes everything in INFO, WARNING, and ERROR plus the following: <ul style="list-style-type: none"> <li>■ Verbose Security</li> <li>■ Display Files</li> <li>■ Client Debug</li> <li>■ Client Connections</li> <li>■ Client Audit</li> <li>■ Client SQL</li> </ul>
INFO	Includes the following information. <ul style="list-style-type: none"> <li>■ Log Jobs</li> <li>■ Checkpoint</li> <li>■ Settings/Configurations</li> <li>■ Job Numbers</li> <li>■ Warnings and Errors</li> </ul>
WARNING	Includes ERRORS.
ERROR	Includes just application errors.

## Formatting and Locating Log Files

You can set the format for your On Screen and Performance Data log files using formatting codes described in the following tables.

### For Example

The **Log to File** setting, \\WD\_LOG\_%Y%m%d\_%H.log, creates a file called "WD\_LOG\_20100922\_11.log" in the Logs | 2010 | 09 | 22 folder at 11:30 AM on September 22, 2010.

The following table contains data from the Microsoft MSDN Library.

Formatting Code	Formatting Description
%a	Abbreviated weekday name
%A	Full weekday name
%b	Abbreviated month name
%B	Full month name
%c	Date and time representation appropriate for locale
%d	Day of month as decimal number (01 – 31)

Formatting Code	Formatting Description
%H	Hour in 24-hour format (00 – 23)
%I	Hour in 12-hour format (01 – 12)
%j	Day of year as decimal number (001 – 366)
%m	Month as decimal number (01 – 12)
%M	Minute as decimal number (00 – 59)
%p	Current locale's A.M./P.M. indicator for 12-hour clock
%S	Second as decimal number (00 – 59)
%U	Week of year as decimal number, with Sunday as first day of week (00 – 53)
%w	Weekday as decimal number (0 – 6; Sunday is 0)
%W	Week of year as decimal number, with Monday as first day of week (00 – 53)
%x	Date representation for current locale
%X	Time representation for current locale
%y	Year without century, as decimal number (00 – 99)
%Y	Year with century, as decimal number
%z, %Z	Either the time-zone name or time zone abbreviation, depending on registry settings; no characters if time zone is unknown
%%	Percent sign

## Performance Data Logging

Performance Data Logging allows you to track jobs in versus jobs out and calculate the rates that jobs are being sent through the Software Print Server.

Performance Logs are Comma Separated Value (CSV) files that contain information on print jobs for the entire Software Print Server system or per printer.

Performance Data	Description
DTM	The date and time marking the end of the logging interval.
TYPE	The data source.
NAME	The printer name.
ELAPSEDMSECS	The number of milliseconds in the logging interval.
INPUTCOUNT	The number of jobs sent to the printer during the logging interval.
INPUTRATEPERSEC	The rate jobs were received per second during the previous logging interval.
INPUTRATEPERMIN	The rate jobs were received per minute during the previous logging interval.

Performance Data	Description
INPUTRATEPERHOUR	The rate jobs were received per hour during the previous logging interval.
OUTPUTCOUNT	The number of jobs printed from the printer.
OUTPUTRATEPERSEC	The rate jobs were output from the printer per second during the previous logging interval.
OUTPUTRATEPERMIN	The rate jobs were output from the printer per minute during the previous logging interval.
OUTPUTRATEPERHOUR	The rate jobs were output from the printer per hour during the previous logging interval.

## Total Jobs

Select **Total Jobs** to create a CSV file containing performance data for all your print jobs.

### For Example

The following is example data from a total job performance data file set to a 30 minute interval.

```
DTM, TYPE, NAME, ELAPSEDMSECS, INPUTCOUNT, INPUTRATEPERSEC, INPUTRATEPERMIN, INPUTRATEPERHOUR,
OUTPUTCOUNT, OUTPUTRATEPERSEC, OUTPUTRATEPERMIN, OUTPUTRATEPERHOUR
"2010/09/23
14:30:00", "JOB", "TOTALJOBS", 1800000, 17, 0.01, 0.56, 34.00, 17, 0.01, 0.56, 34.00
"2010/09/23
15:00:00", "JOB", "TOTALJOBS", 1800000, 14, 0.00, 0.46, 28.00, 14, 0.00, 0.46, 28.00
```

## Per Printer

Select **Per Printer** to create a CSV file containing performance data for each printer.

### For Example

The following is example data from a per printer performance data file set to a 30 minute interval.

```
DTM, TYPE, NAME, ELAPSEDMSECS, INPUTCOUNT, INPUTRATEPERSEC, INPUTRATEPERMIN, INPUTRATEPERHOUR,
OUTPUTCOUNT, OUTPUTRATEPERSEC, OUTPUTRATEPERMIN, OUTPUTRATEPERHOUR
"2010/09/23
14:30:00", "JOB", "PRINTER001", 1800000, 10, 0.01, 0.33, 20.00, 10, 0.01, 0.33, 20.00
"2010/09/23
14:30:00", "JOB", "PRINTER002", 1800000, 7, 0.00, 0.23, 14.00, 7, 0.00, 0.23, 14.00
"2010/09/23
15:00:00", "JOB", "PRINTER001", 1800000, 8, 0.00, 0.26, 16.00, 8, 0.00, 0.26, 16.00
"2010/09/23
15:00:00", "JOB", "PRINTER002", 1800000, 6, 0.00, 0.20, 12.00, 6, 0.00, 0.20, 12.00
```

## Interval

Select the frequency data should be sent to the log files. Each CSV file contains two entries for each printer or job total. The first entry records the input and output data for the previous time interval (30 minutes by default). The second entry records the input and output data from the first entry to when the file was created. So each CSV file contains data for a maximum of twice the logging interval. For example, if the logging interval is 30 minutes, there will be a maximum of 60 minutes of data recorded

in the file. There may be a shorter time period recorded if the Software Printer Server is stopped.

**Note:** Shorter log intervals may affect system performance.

**Note:** When creating formulas to help in analyzing performance data, use the milliseconds recorded in the CSV file rather than the interval you set in the Software Print Server Configuration Utility. LPS adds an entry to the CSV file on or about the logging interval. The actual interval will vary slightly based on this.

## Notification and Status Reporting Tabs

The Status and Notifications tabs are associated with the Client Status Program and the Software Notification Agent. These tabs are disabled if you do not have the Premier Edition of the LPS.

### Related Information

For information on the Status Client and Notification Agent, see the *Software Clients* section of this guide.

## Housekeeping Tab

The purpose of Housekeeping is to allow you to set a time when the levels of disk space used by Printed Jobs and Job errors (in MB) are checked. Printed Jobs and Job Errors are saved into various working folders, and while it is helpful to keep these old job files around for a period of time for auditing purposes, they must be deleted or purged eventually in order to minimize disk space and keep label printing at a high speed. Sections of this tab are described below.

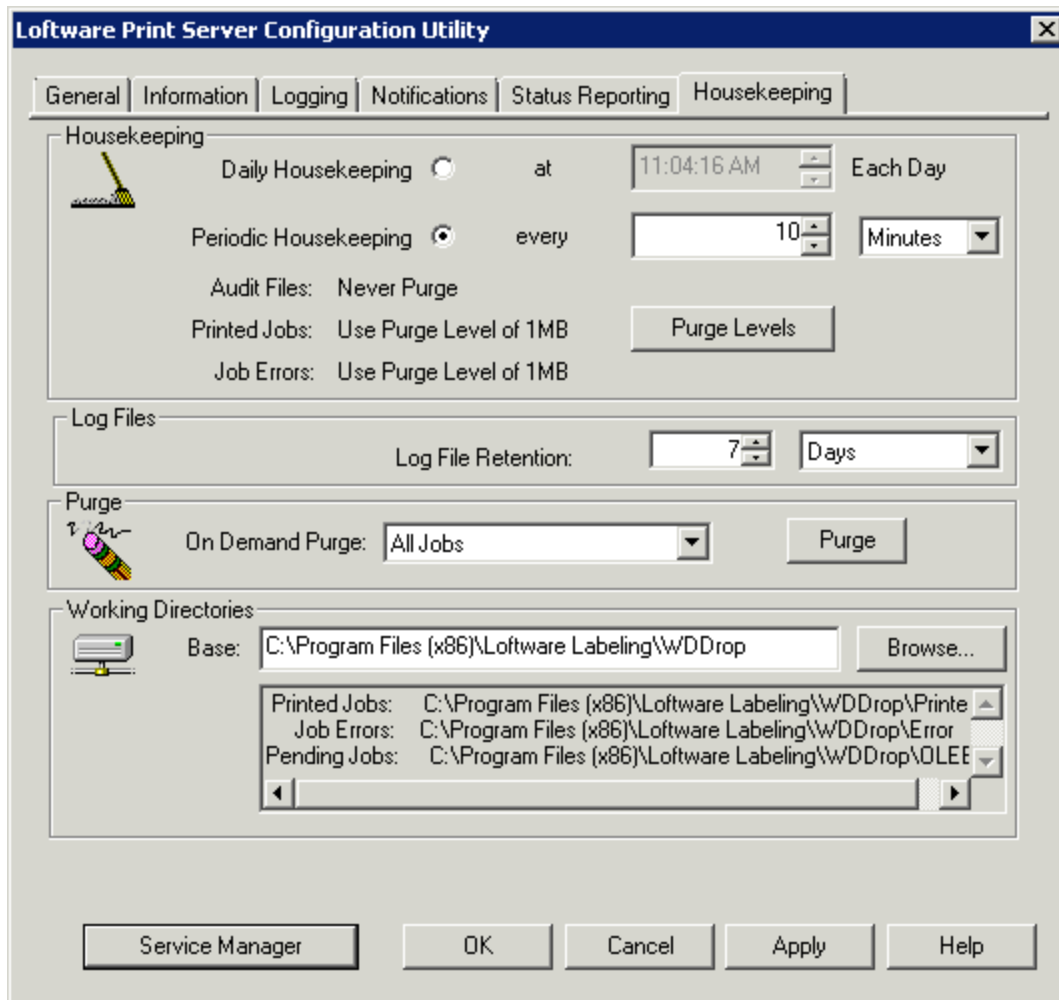


Figure 3.4: Housekeeping Tab

## Housekeeping

### Daily Housekeeping

If Daily Housekeeping is selected, then the Periodic Housekeeping option is disabled. This allows you to set the time of day each day that you would like your Purge Levels to be checked. The editable fields are Hour, Minute, Second, and AM/PM (HH:MM:SS AM/PM). If you run the LPS for 2 consecutive work shifts for example, you may want to set the housekeeping time for 1:00AM, when production is not taking place. The default time is 1:00AM daily.

### Periodic Housekeeping

When this option is selected, housekeeping is performed daily, when the interval time, set in minutes, hours, or days, has been reached. The interval time is set to check the **Purge Levels** of **Printed Job**, **Audit Files**, and **Job Errors** to see if they need to be reset and purged. The default time is 10 minutes. If the Purge Level has been reached, the selected file categories are checked and purged if necessary. The

Periodic Housekeeping time sets when to check if the values set in the drop-down lists for **Audit Files**, **Printed Jobs** and **Job Errors** have been met or exceeded. If they have not, then no purging takes place. The Periodic Housekeeping time can only be set when the Daily Housekeeping option is NOT checked.

## Purge Levels

Saving old jobs is helpful for auditing purposes; however, to avoid running out of drive space, these files must be deleted at some point. You can set this to occur automatically by settings the Purge Level Options for Audit Files, Printed Jobs and Job Errors. If you select the **Use Purge Level** option, when the space consumed by the files in a directory reaches the designated purge level (a number set in Megabytes), disk space is recovered by deleting the oldest files until the size of the directory has been reduced to 75% of the purge level. If the setting is 1 (default), then 250k of the Audit Files, Printed Jobs or Job Errors files are deleted when they reach the 1 MB level.

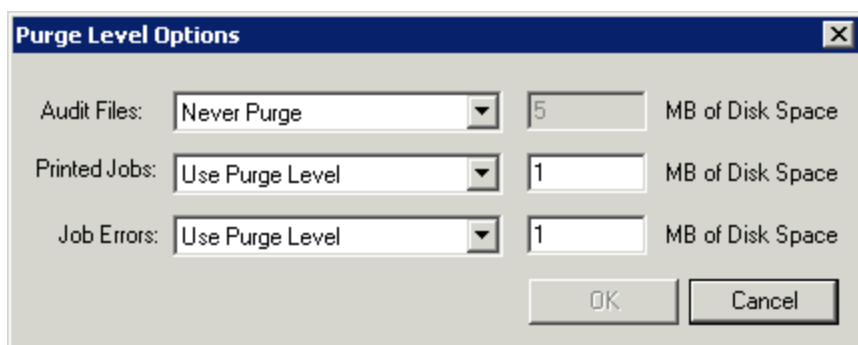


Figure 3.5: Purge Level Options Window

Purge levels are evaluated **ONLY** when the Evaluation Interval time is reached. Because of this, the space actually consumed in the directories may be slightly higher than the purge setting before the purge takes place.

If you set the Purge Level to **Never Purge**, it is your responsibility to delete the files at the appropriate time. If you neglect to do this, you will eventually run out of disk space, which can cause unpredictable issues.

Setting to **Always Purge** deletes all of the files in the directory immediately after processing. Any extra disk space consumed by buffering old files is eliminated; however, you no longer have the ability to view history or job errors and resubmit jobs using the Status program.

## Log Files

### Log File Retention

Set the length of time log files are saved on your server. You can set this for a number of days, weeks, months or years.



## Purge

### On Demand Purge

The drop-down list allows you to select the file types to purge. Select the file type you would like to purge and click **Purge**. Use this option if you need to perform a manual purge.

## Working Directories

By default, the LPS creates three working directories under the <Program Files folder>\Software Labeling\WDDrop.

- Printed
- Errors
- OLEBP

These are the only directories used by the LPS regardless of how many directories you are scanning or the number of printers that you are driving. The **Browse** button is provided if you need to change their locations.

**Note:** One reason to change the Working Directories location is if you want to centralize them for another LPS server to use in an auto-failover situation. In most situations, you will not need to change these directories.

The Software Print Server uses the working directories in the following ways:

- All requests coming through any interface are renamed and saved into the appropriate directory depending on the outcome of the job.
- Jobs originating from the TCP/IP interface are saved in the format specified in the protocol header when submitted. The extension of these files is changed to .int.
- Although the syntax of file interface requests created in .CSV, PAS, and XML are not changed, their names are. The most efficient way to view the files is with the Status program.
- Print requests that have printed without error are saved in the Printed directory along with a corresponding .lpj file. The .lpj files contain information about the job and are for internal system use only. Under normal operating conditions, the Printed directory grows the fastest and is purged according to the housekeeping settings.
- Requests that encounter critical error conditions are saved into the Errors directory. Examples of a critical error are syntax errors in your PAS file or referencing a label that does not exist. Other errors like "Printer out of Labels" are placed in the OLEBP directory and scheduled to retry until the job prints. No user interaction is required for these types of errors.
- Jobs that are ready to print but have not actually been processed by the print engine are buffered in the OLEBP directory until they are sent. This allows the LPS to auto recover any jobs that were not printed after a system crash or unscheduled shutdown.

- The only way to prevent the LPS from auto-recovering is to shut it down and manually delete the jobs in the OLEBP directory.
- Jobs in the Printed and Errors directory can be resubmitted by right clicking them in the Status program and selecting **re-submit**. Any job that is resubmitted is copied to a new name in the printed directory and has an .rsd extension.

#### **Related Information**

For more information on failover, refer to the *Clustering* and *Auto-Failover* sections in this guide.

---

## Label Versioning

Label Versioning is the tracking and numbering of the changes made to individual labels.

### Considerations

- Label Versioning in LPS and LLM is designed to be a single-user system.
  - Files are not locked during editing.
  - There is no automatic merging capability.
- Images cannot be separately version controlled in LPS or LLM, though they can be embedded within labels.
- Label Versioning is a system-level setting.
- Practically, each version of a label is a complete copy of that label stored in a hidden folder. Because of this, the amount of disk space required for labels will increase. This space requirement is greater if you embed images within labels.

See the *Software Label Manager User Guide* for more information on using Label Versioning and embedding images.

### Using with Existing Labels

With Label Versioning on, when you save previously created labels, the version number increments from the last version saved. For example, if you saved five revisions of a label prior to turning on Label Versioning, when you next save the label, the version number will be six. You can view information associated with the previously created version from the Label Setup and Properties dialog.

### Promotion

Promotion is the act of moving a label from one version-controlled system to another version-controlled system. One example is moving a new label into a Production environment (LPS) from a Development environment (Design).

In order to promote a label, you must be logged into the system you are promoting from as a user with access to the folder to which you are promoting. Typically this means that the user promoting should be a member of the `Loftware_Group` user group on the system to which the label is being promoted.

A command line utility, [LWVersionUtility](#) is available. This utility may help you automate the promotion process.

## Instances

Instances are the different hardware/software environments that you use to create, manage, and print labels. With Label Versioning on, you Promote labels to move them from one instance to the next. Typical instances or environments include:

- **Development** - where labels are created and customized.
- **Test** - where new and upgraded labels are tested with sample data. A test environment may also be used to test the Promotion process.
- **Quality** - where labels and process are tested with data, software, hardware and configurations similar to that of the Production instance.
- **Production** - where final labels are made available for printing.

**Note:** You can configure and use up to 20 instances.

## Instance Checking

You can restrict labels from printing on instances other than one the label is configured for.

If you create labels with Label Versioning on and then change the InstanceName, if instance checking is on, you will no longer be able to print the labels saved with the previous Instance Name from the current environment. You must open and save the label after the Instance Name change or Promote a new version from a different instance.

### For Example

When Label Versioning is turned on, the current instance (InstanceName) is called Default. If you change the name to Development, any labels created while the InstanceName was Default will not be printable until they are reopened and saved or a new version is promoted.

## Enable Label Versioning

Label Versioning is turned on and configured using the LWVersion.ini file.

1. Access the configuration file from the **Start** menu under Software Labeling.  
By default: C:\ProgramData\Software Inc\Labeling\Config
2. Open LWVersion.ini.
3. Change AllowLabelVersioning to 1
4. If you want the configured instance of a label to be checked against the instance the label is being printed on, set InstanceNameChecking to 1.
5. Name the current instance by entering an InstanceName.
6. Define the Instances in your configuration by entering instance names and paths under [Instances].

7. Uncomment (remove the semicolon) the [Instances] heading and any configured instances.
8. Save the file.

### For Example

The following file would configure a system to allow Label Version and enforce Instance Name Checking. The system would be called Development, and labels from the system would be promotable to a Testing and a Production environment.

#### Example LWVersion.ini

```
; Set AllowLabelVersioning to 1 to enable Label Versioning
[Version]
AllowLabelVersioning=1

; Set InstanceNameChecking to 1 to validate the label instance against the
; local instance before printing.
; You might use this to prevent Development labels from printing on
; Production.
; InstanceNameChecking setting will be honored only when
; AllowLabelVersioning is set to 1
InstanceNameChecking=1

; InstanceName is used to identify the local instance.
; For example, the LLM/LPS in a test environment would be identified by
; InstanceName=Test
; Change the following InstanceName from Default to the appropriate name for
; this instance.

[Instance]
InstanceName=Development

; Instances provide the InstanceName and label path to which labels from thi
; instance can be promoted.
; These names will appear on a sub menu when you select File | Promote.
; Instances is NOT typically set in on a Production (final destination)
; instance.
; Uncomment the following example, change the name and path, and add any
; additional instances to which this
; instance can promote.

[Instances]
Testing=\\LPS_Server_Test\software$\LABELS
Production=\\LPS_Server\software$\LABELS
```

This page intentionally left blank

Once you have successfully installed and configured the Software Print Server, you can start it and make a test request through one of the interfaces. You can start LPS in one of these modes:

- Service mode
- Interactive mode

### Service Mode

The Software Print Server is designed to run as a Windows service. A service application is a long-running application that can be configured to run in its own Windows session when the server is started. A service does not require user intervention, nor does it have a user interface. Services therefore do not interfere with other activities or users on the same computer or server, and they are protected from unintentional changes.

Two methods can be utilized to start the Service. Each method allows you to set the Service to start automatically at system boot, so that you do not have to remember to start it each time your system is rebooted. The Service defaults to manual start. Be sure to run the **Software Print Server Configuration Utility** before starting the service to configure it properly. Don't switch to auto start until you have finished verifying your system. The reason for this is you may want to flip back and forth between Service and Interactive modes while testing and debugging your system.

**Note:** Since Services cannot display error or status information on your desktop, starting and testing your labeling system in Interactive mode is recommended during testing, development, and troubleshooting. When your system is in production, make sure you are logging critical error information to the Event Log.

### Start the LPS in Service Mode

#### To start the LPS Service

1. Open the **Software Print Server Configuration Utility** (LPS Configuration) from the **Start** menu under Software Labeling.
2. Click **Service Manager** in the LPS Configuration Utility.
3. Select **Software Print Server** from **Services**, and click **Start**. The service is started when the icon turns green and the **Start** button becomes disabled.
4. Click **Close** to hide the **Service Manager** window. The service continues to run.

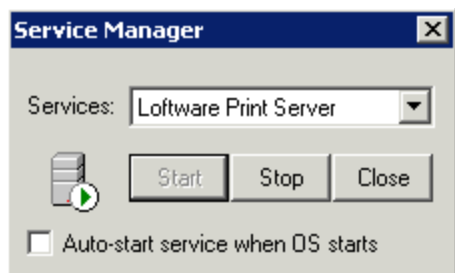


Figure 3.1: Service Manager with Service Started

## Interactive Mode

When the LPS is run in interactive mode, the **Software Print Server** status window allows you to monitor what the LPS is doing at any given moment. You can also see which client applications are interacting with the LPS. The **Options** menu has a **Security Checks** selection that performs a diagnostic scan of the LPS installation. This is helpful for troubleshooting any problems you may encounter while setting up your system. Use the **Logging** tab of the **Software Print Server Configuration Utility** to turn on extra debugging information that can be seen on the Software Print Server status window while running in this mode.

**Important:** You will typically turn logging off on productions systems. Production systems should be run as a service.

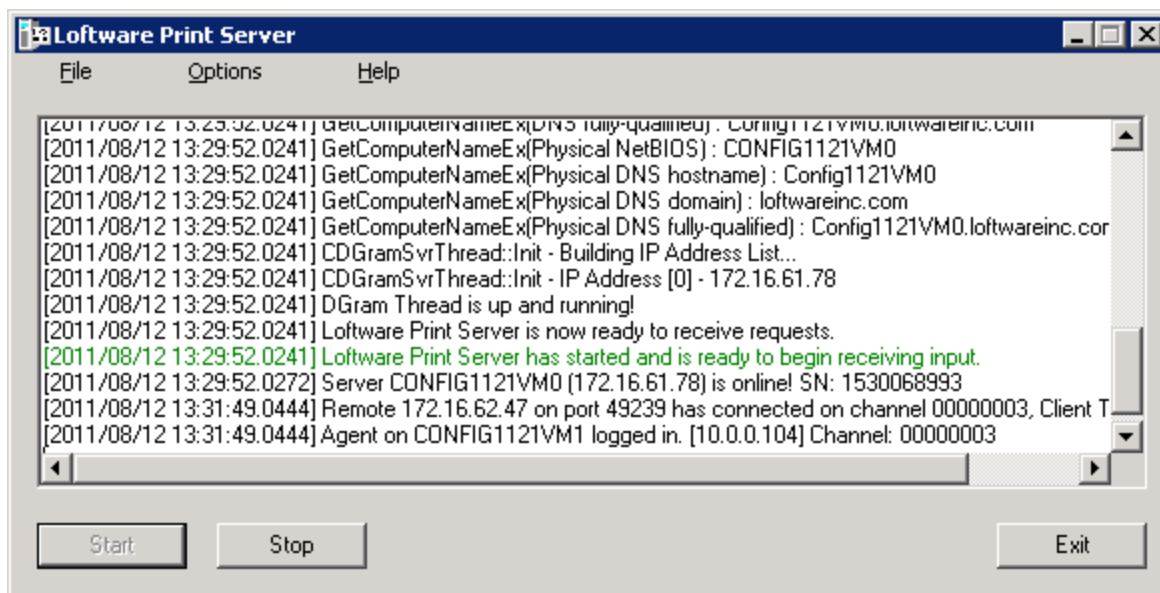


Figure 3.2: LPS Running Interactively

**Note:** If you are running the LPS as a Service, you must stop the Service before running the LPS interactively. You may want to run in Interactive mode to perform a diagnostic scan or take advantage of On Screen Logging.



- To start the LPS, select LPS Interactive from the **Start** menu under Software Labeling.
- To perform a diagnostic scan of your LPS installation, select Options | Security Checks. This may help you troubleshoot issues you may encounter while setting up your system.

You must click **Start** for the LPS to begin accepting data from its interfaces. If you launch the LPS with a “-run” parameter, it automatically starts.

This page intentionally left blank

---

## Print Request Data Structures

There are three data structure syntax which can be used to make requests to the Software Print Server, PAS, CSV, or XML.

Any of these data structures can be used regardless of the interface being used unless otherwise noted in the interface section.

First, the front end application determines the following.

- Which label template to print
- What data to place in the template
- Label quantity
- Target printer

The application will then place this data into a PAS, CSV, or XML structure and send it to the LPS through the interface. Many of the examples in this section assume that the File Interface is being used. The examples still apply to the other interfaces, you would just send the data structure through the interface (like a socket) instead of creating a file.

**Note:** If you wish to modify any of the Software default XSD or DTD schema files that ship with the product, create and modify a renamed copy of the file. Otherwise, any subsequent install may overwrite changes you made.

### PAS and CSV Commands

All LPS PAS commands start with an asterisk (\*). Each PAS file must start with the \*FORMAT command, and end with the \*PRINTLABEL command. The following commands apply to the PAS and CSV file syntaxes. The commands for XML are the same with the exception of starting with an '\_'.

LPS Command	Example	Description
*FORMAT	*FORMAT, Label1.lwl	The _FORMAT command references the label format to use. If no path is specified, the LabelsPath entry in Options   File Locations is used. If the file is missing an extension, the default '.lwl' is assumed.
*JOBNAME	*JOBNAME, SampleJob001	Optional entry used to identify the current job during the printing process. If _JOBNAME is not present, the filename is used.
*QUANTITY	*QUANTITY, 1	Optional entry used to designate the quantity of labels for print. If _QUANTITY is NOT present, a default of 1 (one) is assigned.

LPS Command	Example	Description
*DELINKODBC	*DELINKODBC	Overrides the database, and you must provide all data for the labels. This is helpful for when you normally 'push' the data to the LPS for your production, but the fields are connected to a database for backup purposes using the On Demand Print Client. Include this command in your PAS file to turn off the database links for normal production.
*DUPLICATES	*DUPLICATES, 1	Optional entry used to designate the number of duplicate labels for print. If there are any Incrementing/Decrementing or Serial number fields on the label, _Duplicates will cause duplicate serial numbers.
*PAGES	*PAGES, 1	Optional entry used to designate the number of pages to print when a label has a layout. There is no effect if no layout is used.
*PRINTERNUMBER	*PRINTERNUMBER, 1	Required entry used to designate the printer to which the label is printed. The *PRINTERNUMBER value corresponds to the Printer number in Configure Printers dialog. Or use PRINTERNAME to identify the printer with an alias name.  <b>Note:</b> If you move a printer to a different line in the LLM Device Configuration grid, you need to update your PAS file with the new number. Consider using PRINTERNAME if you will be changing the order of printers.
*PRINTERNAME	*PRINTERNAME, PrinterAlias	Where PrinterAlias is the Alias name assigned to the printer in Configure Printers Connection dialog.  Only one entry, PRINTERNUMBER or PRINTERNAME is needed.
*PRINTLABEL	*PRINTLABEL	Prints the current label. Any commands after this are considered a new label. Your label will not print without this required command.
*SENDFORMAT	*SENDFORMAT, 0	Optional entry. Value of <b>1</b> causes LPS to suppress the sending of the format information to the printer. The printer recalls the format from memory and the throughput time is increased. This only works on printers that support the pre-downloading of the fixed fields on your format.  Value of <b>0</b> turns this option off, which is the default setting.
*TRAY	*TRAY, 1 *TRAY, lower	Optional entry used to designate the tray to print to when printing using PCL5, PCL5c or Windows drivers. Tray data corresponds to the choices in the printer driver's properties box. For Windows drivers use the tray name (not case sensitive). For PCL5 and PCL5c use the tray ID number.

LPS Command	Example	Description
[Options] StickyTray	[Options]Stickytray=1	Sets a specific printer tray. The tray data must match the selections in the printer driver's Properties box.
*VERSION	*VERSION, 1	Optional entry used to specify the version of the label to print. The head revision is printed if no version is specified. Specifying a version that does not exist will cause an error.

## PAS and CSV Syntax

The PAS syntax is a simple ASCII structure that has all the commands, field names, and data for the labels that are to be printed. The field names in the file match the ones given to their corresponding fields on the label during design time. Each PAS file can have many different label requests within it. Once constructed, this file is passed to the LPS through the interface, which is usually a file or a socket.

### Notes on using the \*Tray command

To select a tray, send a \*TRAY command in the PAS file. For example:

```
*TRAY,1
```

To select a tray in a CSV file, send the TRAY command in the header, and specify the tray name or ID in the record request/line. For example:

```
"*PRINTERNUMBER", "*QUANTITY", "*FORMAT", "*DUPLICATES", "*TRAY",  
"Field1", "Field2"  
5,1,LabelName,1,1,Name,Product  
5,1,LableName,1,4,Name,Product
```

**Note:** The \*TRAY command is NOT retained by default; in other words, if you send a stacked PAS with a \*TRAY command in the first request, subsequent requests do NOT print to the same tray (unless the default tray was selected), as it is OFF or set to 0 by default. To retain the \*TRAY command, set the following in the LLMWDN32 Configuration file (**Start** menu under Software Labeling).

### Alternative Commands

If you are using the alternative syntax, you may use the following commands as alternatives to the previously mentioned commands. The alternative syntax makes it easier for report generation software to create the file.

**Note:** Use uppercase letters when specifying alternative commands.

LPS Command	Alternative Syntax
*FORMAT	FORMAT
*PRINTERNAME	PRTNAME

LPS Command	Alternative Syntax
*QUANTITY	LLMQTY
*PAGES	PAGE
*JOBNAME	JOBNAME
*PRINTERNUMBER	PRTNUM
*DUPLICATES	LLMDUP
*VERSION	LWVERSION (see Note below)

**Note:** The \*PrintLabel command has no alternative because it is not used in the CSV syntax.

**Note:** The \*VERSION alternative command should not be "VERSION" because VERSION is a reserved word as of LPS version 10.0 and higher. Using VERSION may cause printing problems and the need for Loftware Technical Support assistance.

**Note:** Null characters are not allowed in PAS files. The label may print as expected, but the file may not be completely visible from the File Editor in the Loftware Status Dialog (Client). This will cause the Resend function to fail.

**Note:** If the previously mentioned alternative commands do not work for you, you may specify others by creating a section in the LLMWDN32 Configuration file (**Start** menu under Loftware Labeling) called [CommandRemap]. In this section you may specify:

```
[CommandRemap]
FORMAT = yourFormatCommand
*PRINTERNAME = yourPrinterNameCommand.
```

## Example 1: Simple PAS File

This file loads a format called NAMETAG.LWL that was previously designed in Loftware Label Manager. The next two lines specify data for variable fields that you named NAME and COMPANY when the label was designed. The next lines instruct the LPS to print two of these labels on configured printer #3.

### Example using LPS PAS commands

#### Simple.pas

```
*FORMAT, NAMETAG.LWL
NAME, John Smith
Company, ABC Company
*QUANTITY, 2
*PRINTERNUMBER, 3
*PRINTLABEL
```

## Example using Alternative commands

### Simple.pas

```
FORMAT, NAMETAG.LWL
NAME, John Smith
Company, ABC Company
LLMQTY, 2
PRTNUM, 3
*PRINTLABEL
```

## Example 2: Stacked Request PAS Files

This example illustrates that you can stack requests in a PAS file. It also shows how you might use the \*JOBNAME and \*PRINTERNAME commands. If you need to stack more than 50 requests, your file is much smaller using the alternate syntax following this example.

## Example using LPS PAS commands

### Stacked\_request.pas

```
*FORMAT, NAMETAG.LWL
NAME, John Smith
Company, ABC Company
*QUANTITY, 2
*PRINTERNAME, Human Resources
*PRINTLABEL
*FORMAT, AIAG.LWL
PARTNUM, A100
LOTNUM, 3400
SERIALNUMBER, S000001
SUPPLIER, ACME1.78922
*QUANTITY, 10
*PRINTERNAME, Shipping Printer 4
*PRINTLABEL
*FORMAT, AIAG.LWL
PARTNUM, BG500
LOTNUM, 00022
SERIALNUMBER, S000002
SUPPLIER, ACME1.78922
*QUANTITY, 1
*PRINTERNAME, Receiving Printer2
```

## Example using Alternative Commands

### Stacked\_request.pas

```
FORMAT, NAMETAG.LWL  
NAME, John Smith  
Company, ABC Company  
LLMQTY, 2  
PRTNAME, Human Resources  
*PRINTLABEL  
FORMAT, AIAG.LWL  
PARTNUM, A100  
LOTNUM, 3400  
SERIALNUMBER, S000001  
SUPPLIER, ACME1.78922  
LLMQTY, 10  
PRTNAME, Shipping Printer 4  
*PRINTLABEL  
FORMAT, AIAG.LWL  
PARTNUM, BG500  
LOTNUM, 00022  
SERIALNUMBER, S000002  
SUPPLIER, ACME1.78922  
LLMQTY, 1  
PRTNAME, Receiving Printer2
```



### Example 3: .CSV Files

Stacked Request PAS Files can also be represented in ASCII CSV file format with header as:

```
"FORMAT","PRTNAME","JOBNAME","LLMQTY","NAME","COMPANY","PARTNUM", +
"LOTNUM","KANBAN","DATE","SERIAL NUMBER",SUPPLIER"

"NAMETAG.LWL","Human Resources","LabelJob1","2","John Smith", +
"ABC + Company","","","","","",""

"AIAG.LWL"Shipping Printer 4","","10","","","A100","34000","RS200", +
"1/10/00","S000001","ACME1.78922"

"AIAG.LWL","Receiving Printer 2","","1","","","BG500","00022","KLJ4500", +
"3/12/98","S000002","ACME1.78922"
```

**Note:** The + at the end of the line indicates that this line is continued on the next line. Your file does not contain the plus or wrap to the next line. You only need to start a new line when you are starting a new label.

**Note:** The data fields not used by a particular label are empty (""). Quotation marks are not necessary for the data, but if your data contains commas, quotation marks are mandatory. Also notice that the previous example uses the alternate commands for \*PRINTERNAME etc. and that the alternate syntax is much more compact.

```
"FORMAT","PRTNUM","LLMQTY","NAME","COMPANY"
"NAMETAG.LWL","3","2","JOHN SMITH","ABC COMPANY"
"NAMETAG.LWL","2","1","JANE DOE","X COMPANY"
"NAMETAG.LWL","3","20","BOB JOHNSON","ABC AGENCY"
```

**Note:** For .csv files, each new record (label) is added on a new line. The header on the top line is only needed once. The example shows multiple record requests in one file. Each label uses the same label format but prints to a different configured printer and has different data. Notice that the alternative command names are used in the header record (line 1).

## XML Files

### XML Commands

All commands begin with an underscore '\_'. Commands are sticky which means that the XML header defaults are always used unless a request header overwrites any of them later in the XML file.

Creating a stacked (multiple requests per file) XML file for one particular label format only requires one `_FORMAT` command entry in the XML header.

**Note:** Use uppercase letters when specifying XML commands.

XML Command	Example	Description
<code>_FORMAT</code>	<code>_FORMAT="Label1.lwl"</code>	The <code>_FORMAT</code> command references the label format to use. If no path is specified, the <code>LabelsPath</code> entry in <code>Options   File Locations</code> is used. If the file is missing an extension, the default <code>'.lwl'</code> is assumed.
<code>_JOBNAME</code>	<code>_JOBNAME="SampleJob001"</code>	Optional entry used to identify the current job during the printing process. If <code>_JOBNAME</code> is not present, the filename is used.
<code>_QUANTITY</code>	<code>_QUANTITY="1"</code>	Optional entry used to designate the quantity of labels for print. If <code>_QUANTITY</code> is NOT present, a default of 1 (one) is assigned.
<code>_DUPLICATES</code>	<code>_DUPLICATES="1"</code>	Optional entry used to designate the number of duplicate labels for print. If there are any Incrementing/Decrementing or Serial number fields on the label, <code>_Duplicates</code> will cause duplicate serial numbers.
<code>_PAGES</code>	<code>_PAGES="1"</code>	Optional entry used to designate the number of pages to print when a label has a layout. There is no effect if no layout is used.
<code>_PRINTERNUMBER</code>	<code>_PRINTERNUMBER="1"</code>	Required entry used to designate the printer to which the label is printed. The <code>_PRINTERNUMBER</code> value corresponds to the Printer number in <code>Configure Printers</code> dialog. OR...
<code>_PRINTERNAME</code>	<code>_PRINTERNAME="PrinterAlias"</code>	Where <code>PrinterAlias</code> is the Alias name assigned to the printer in <code>Configure Printers Connection</code> dialog. Only one entry, <code>PRINTERNUMBER</code> or <code>PRINTERNAME</code> is needed.
<code>_TRAY</code>	<code>_TRAY="1"</code>	Optional entry used to designate the tray to print to when printing using PCL5, PCL5c or Windows drivers. Tray data corresponds to the choices in the printer driver's properties box. For Windows drivers use the tray name. For PCL5 and PCL5c use the tray ID number.

XML Command	Example	Description
_VERSION	_VERSION="1"	Optional entry used to specify the version of the label to print. The head revision is printed if no version is specified. Specifying a version that does not exist will cause an error.

## XML File Syntax

XML files that are sent to the Software Print Server must begin with the following.

```
<?xml version="1.0" standalone="yes"?>
```

## XML Header

The XML Header is a beginning tag in which you define the default commands for the entire XML file.

- The Header starts with <labels and is followed by any default commands you wish to set separated with a space.
- End the header with a >. If no commands are set within the header, then they must be set in the beginning tag of each individual request.
- The XML namespace (xmlns) is needed if the XML will be validated against the XML Schemas (XSD).

### Example XML Header

```
<labels _FORMAT="Label1.lwl" _JOBNAME="SampleJob001" _QUANTITY="1" _  
PRINTERNUMBER="1" xmlns="http://www.loftware.com/schemas">
```

OR- without any commands

```
<labels>
```

## XML Requests and Commands

- Requests can be in any order, first entry per request is a request header <label>
- You may overwrite any of the default commands set in the XML header or add more commands to this one request.
- If no commands are set here, the defaults are used.
- The final entry per request is a </label> line
- Multiple requests are possible and are separated by the end of one request </label> and the beginning of the next request <label>.

**Note:** When adding commands, separate each with a space. Also note that any default commands overwritten here are for this one request only.

### Example XML Request Header

```
<label>
```

The previous would use the defaults listed in the XML header -OR-

```
<label _QUANTITY="5" _PRINTERNUMBER="2">
```

This one request would use `_FORMAT` from the XML header but its own `_QUANTITY` and `_PRINTERNUMBER`.

## XML Data

All data lines begin with `<variable name=` followed by the field name in quotation marks followed by `'>` then the data for the field and finally an end tag `</variable>`.

If a field does NOT appear in the XML file, the data is cleared and NOT remembered across labels.

### Example XML Data Line

```
<variable name="Text000">New Data</variable>
</label>
```

Marks the end of the current request definition.

```
</labels>
```

Marks the end of all requests in the entire XML file.

**Note:** If you are unsure if you have written your XML file correctly, open the XML file using an Internet browser. If the file opens without error, your XML file is valid, and is processed when dropped to the scan directory.

## XML Validation

The Software Print Server can validate XML against XML Schemas (XSD file). By default the Software Print Server does not validate XML.

### Legacy DTD Support

The Software Print Server switched to XML Schema validation with the Software Print Server version 10.0. If you want or need to validate your XML, you must follow the steps in [Enable XML Schema Validation](#).

**Important!** This change may cause XML files with references to the Software label.dtd file to fail to print, if you enable XML Schema validation.

**Important!** The Software Print Server does not validate against the DTD file. Even with validation enabled, an XML job that violates the DTD could still print successfully, and no errors will appear.

If you are upgrading to the current version of the Software Print Server from a pre-10.0 version, you can continue to reference the legacy DTD in your XML. By default DTD declarations are allowed, but the reference is ignored, and the XML is not validated.

## Enable/Disable XML Schema Validation

The Software Print Server can validate the XML you send through the print server against the XML Schema (XSD) files included with the Software Print Server.

## Enable XML Schema Validation for XML Print Jobs

By default XML files are not validated. In order to use validation you must enable it in a configuration file. Use the following procedure to enable XML Schema validation for XML files that you drop to the Software Print Server.

**Important:** This procedure involves changing configuration files used by the Software Print Server. Create backups and take care to not make changes to other settings in the file.

1. Locate the LLMWDN32 configuration file from the **Start** menu under Software Labeling, and create a backup copy of the file.
2. Open the LLMWDN32 configuration file in a text editor.
3. Add the [JobHandlingXmlSchema] setting to the file on a new line at the end of the file. Include the brackets.
4. On a new line directly under the [JobHandlingXmlSchema] setting, add Validation=1.
5. On a new line under Validation=1, add ProhibitDTD=1. This excludes the use of DTD references in the XML.

### For Example

```
[JobHandlingXmlSchema]
Validation=1
ProhibitDTD=1
```

6. Restart the Software Print Server.
7. Remove the DTD declaration from your XML files if it exists.

```
<!DOCTYPE labels SYSTEM "C:\Program Files (x86)\Software Labeling\Batch\label.dtd">
```

8. Include the namespace reference in the XML files that you send to the Software Print Server.

```
xmlns="http://www.software.com/schemas">
```

## Disable XML Schema Validation for Scripts

By default Script data sources are validated. Use the following steps to disable script validation.

**Important:** This procedure involves changing configuration files used by the Software Print Server. Create backups and take care to not make changes to other settings in the file.

1. Locate the LLMWDN32 configuration file from the **Start** menu under Software Labeling, and create a backup copy of the file.
2. Open the LLMWDN32 configuration file in a text editor.
3. Add the [ScriptXmlSchema] setting to the file on a new line at the end of the file. Include the brackets.
4. On a new line directly under the [ScriptXmlSchema] setting, add Validation=0.
5. On a new line under Validation=0, add ProhibitDTD=1. This excludes the use of DTD files.

**For Example**

```
[ScriptXmlSchema]
Validation=0
ProhibitDTD=1
```

- Restart the Software Print Server.

## If the Software Print Server is Configured to use Windows Registry Settings

If your Software Print Server is configured to use settings in the windows registry you must perform the following additional steps to enable XML Schema Validation.

- Open the **Software Print Server Configuration Utility** (LPS Configuration) from the **Start** menu under Software Labeling.
- Press Shift | F2. The **Advanced** tab appears. Select the **Advanced** tab.
- Clear the **Remap Ini Files to Registry** checkbox, and click **OK**.
- Add the [JobHandlingXmlSchema] and [ScriptXmlSchema] setting to the LLMWDN32 configuration file as described in Enable XML Schema Validation for XML Print Jobs.
- Repeat steps 1 and 2, and then reselect the **Remap Ini Files to Registry** checkbox. Click **OK**.

### Example 1: XML File

In this example, one label with NAMETAG.LWL's format is printed to Printer 1. The data for fields NAME and COMPANY are filled with "John Smith" and "ABC Company" respectively.

```
<?xml version="1.0" standalone="yes"?>
<labels _FORMAT="NAMETAG" _QUANTITY="1" PRINTERNUMBER="1"
xmlns="http://www.loftware.com/schemas">

<label>
<variable name="NAME">John Smith</variable>
<variable name="COMPANY">ABC Company</variable>
</label>
</labels>
```

### Example 2: Multiple Label XML File

This example prints three labels to Printer 1. The first two labels use the UNIT\_LABEL label format and the last one overrides the header with a different format called PACKLIST. Subsequent labels will revert back to the header if the \_FORMAT command is not overridden again. It is possible to stack thousands of different label requests targeted to hundreds of different printers in a single XML file using this technique.

```
<?xml version="1.0" standalone="yes"?>

<labels _FORMAT="UNIT_LABEL.lwl" _QUANTITY="1" PRINTERNUMBER="1"
xmlns="http://www.loftware.com/schemas">

<label>
<variable name="NAME">Alternator</variable>
<variable name="ADDRESS">26 Josia Way, Door 2</variable>
```

```

<variable name="WEIGHT">23 kgs</variable>
<variable name="KANBAN">789-324434432221</variable>
<variable name="LOT">493820208383</variable>
</label>
<label>
<variable name="NAME">Fan Belts</variable>
<variable name="ADDRESS">78 Hilltop Drive</variable>
<variable name="WEIGHT">5 kgs.</variable>
<variable name="KANBAN">789-234324</variable>
<variable name="LOT">493820208383</variable>
</label>
<label _FORMAT="PACKLIST.LWL">
<variable name="PO">9089898889</variable>
<variable name="DATE">4/22/04</variable>
<variable name="LINE_1">2 Alternator Parts $34.98</variable>
<variable name="LINE_2">17 Fan Belts $123.49</variable>
<variable name="LINE_3"></variable>
<variable name="LINE_4"></variable>
<variable name="TOTAL">$158.47</variable>
<variable name="FINISH_CODE">SHIPPED TODAY...APPROVED</variable>
</label>
</labels>

```

This page intentionally left blank



---

## Maximizing System Flexibility, Minimizing Maintenance

---

The Software Print Server allows you to set up your labeling system to handle changes, additions, and reconfiguration. By following a few rules during setup, flexibility can be achieved:

- Existing Labels can be modified without changing code. This is accomplished by writing your code to extract the variable field names from the label file before each print. If fields are removed or added, your code takes care of it “on-the-fly.” This can be done with the ActiveX and .NET programming controls for computer applications. Host applications can achieve this by using the socket interface. This technique can be circumvented by just sending all data fields in each request; the label will only access the data that pertains to it.
- New Labels can be added without changing code. By constraining the list of available field names in Software Label Manager, designers are only able to use variable fields that your Front End program knows how to handle.
- New Printers can be added from different manufacturers. When printer manufacturers and models change, simply re-save your label with the new model number. The Front-End application need not know anything about this. Depending on how different the printer is, some label design modification may be needed. See About Printer Family Drivers in the *Software Label Manager User Guide* for more information on sharing labels among printers.
- Serialized fields, variable graphics, EPC RFID data and special check digits can be added without changing code. Anything that is added or changed in the Design automatically works.
- Tables and Fields can be added to databases and used in new label formats without having to write code. By keeping the constrained field list up to date with database table and field names, users can change their database and their labels without the knowledge of the Front-End program. The following sections explain how this is done.

### About Data Push

Although the Software Print Server has the ability to access data through 32-bit ODBC drivers, this may not be the most efficient approach. Chances are that your existing application already has access to the data. Because of this, it may be easier and faster for your application to push the data to the LPS. Pushing data to the LPS may help with the following.

- There usually is no need to install ODBC drivers on the LPS server and configure the labels to retrieve data from across the LAN or WAN.
- If your data is on a UNIX system, having to go through connectivity software may affect performance.
- Data Push is meant to simplify your system in setup, performance, and maintainability.

There are cases where you must require the LPS to pull the data from your database. For example, if your application is running on an RF hand-held device that does not have access to the data.

As a best practice, only pull the data into LPS when there is no other choice.

## Providing Data

You can pass fields in your print request file that the label does not print. This can be used to provide your user with a list of legal field names that you include in the drop file which users may design new labels for. These labels work with your system, preventing the need to change the source code. The label called out using the \*FORMAT command pulls only the fields it needs from the drop file, ignoring the rest.

## Constraining Label Fields

The Software Print Server makes provisions for your source application to know which variable data fields are present in your label formats. This allows your application to decide what data to pass through the interface based on which format is being processed. This avoids having to hard code your system for specific labels. Users are able to make changes and add new labels to the system without having to involve programming staff.

### Example

ABC Corporation has a relational database on their host system named BigDatabase. It has two tables with several fields as shown below:

Customer	Product
Name	Part Number
Address	Color
City	Description
State	Weight
Zip	

Using Notepad or another text editor, create the following file, and place it in the <Program Files folder>\Software Labeling directory. In this example, we have chosen to use the "TABLE.Field" convention for later use in an SQL statement.

Name the file l1mfield.lst, and save it in the Software Labeling folder. If there is more than one .lst file, you can choose the appropriate one from the **Advanced** tab in the **Label Setup** dialog box. If there is only one .lst file, then it is chosen by default.

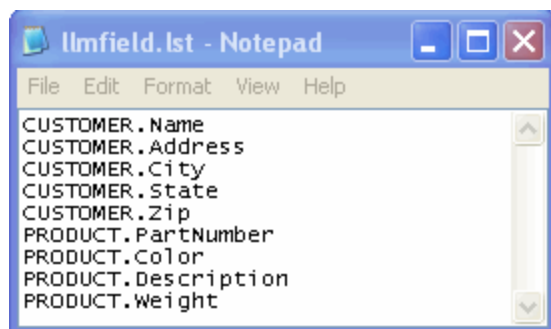


Figure 5.1: A Constrained Field List in a .lst file

The figure below shows how the Properties box behaves when a label is constrained with an .lst file. The **Properties** box constrains the **Field Name** list. In this mode, a name cannot be typed in. It must be chosen from the list, thus preventing a designer from creating a field that cannot be populated by your program.

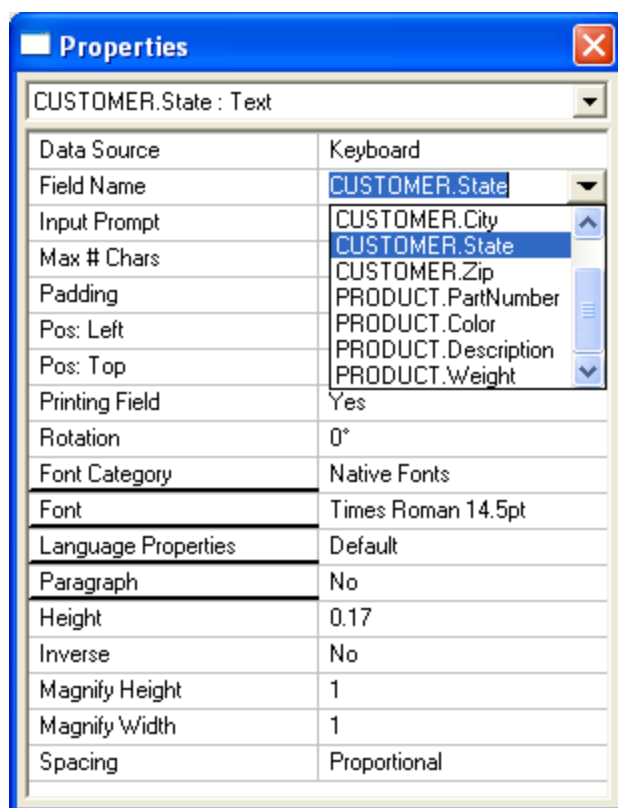


Figure 5.2: Properties box showing constrained field name list

#### Related Information

For information on constraining fields, refer to the *Customizing Labels* section of the *Software Label Manager User's Guide*.

## Getting Field Names from the Label at Print Time

You can use a TAB file to retrieve the names of the fields on a label at print time.

### Enable TAB File

1. Select Options | Preferences.
2. Expand **Design Options** and select **Create Tab file on Save**.
3. Click OK, and restart Design32.

When the label is saved, a file is saved with the same name as the label and a .TAB extension.

#### For example

Suppose you designed a label constrained as described in [Constraining Label Fields](#). The name of the label is LABEL1.LWL and contains fields for CUSTOMER.Name, PRODUCT.PartNumber and PRODUCT.Color. The label and corresponding tab file looks like the following:

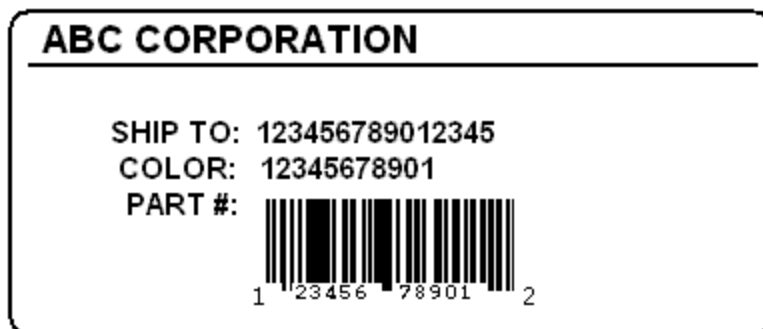


Figure 5.3: Sample Label

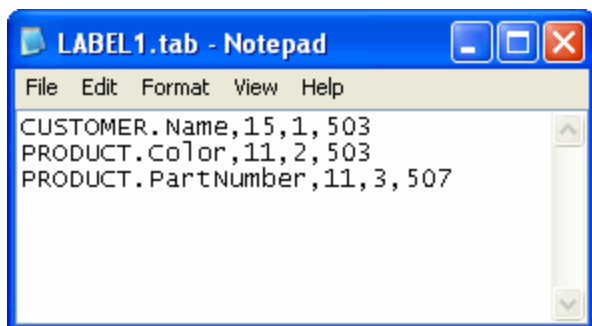


Figure 5.4: TAB file contents for sample label

**Note:** The .TAB file contains field name, field length, tab order, and a field type code. 503 is for variable text fields and 507 is for variable barcode fields. It is likely that the only data your application is interested in is the field name.

If your program runs on a host computer, it needs access to the .TAB file in order to know what fields are on the label. If you are using the socket interface, this information can be obtained through the socket connection rather than the tab file. If your application runs on a computer, it may use either the Software ActiveX control or .NET control to obtain the same information from the .LWL file. This technique is nice, because you can always be sure that the tab order and the label file are synchronized.

Having this information available allows your program to retrieve and supply only the minimum data necessary to print the label. If you choose to include extra data in your request, only fields in the .tab file will be accessed; the others will be ignored.

### Example (continued)

The ABC Corporation has written a Visual Basic Client program that determines through various criteria what labels to print. The Visual Basic program uses the Software ActiveX Client control to get the fields in the label and build a SQL query on-the-fly in order to retrieve data from the UNIX database.

### Example Code

```
"SELECT CUSTOMER.Name, PRODUCT.Color, PRODUCT.PartNumber FROM BigDatabase WHERE
CUSTOMER.Name = 'Smith, John';"
```

### Actual Visual Basic Code Using the ActiveX Client Control:

```
`open the database
Set myDatabase = OpenDatabase(App.Path & "\sample.mdb")
`build SQL statement to grab data for this label
sqlStatement = "SELECT "
ClientX1.SetLabelName "mytest.LWL"
For i = 0 To ClientX1.FieldCount - 1
    sqlStatement = sqlStatement & "[" & ClientX1.FieldName(i) & "]" & ", "
Next i
`get rid of the last comma before the FROM clause and append key
sqlStatement = Left(sqlStatement, Len(sqlStatement) - 2) & " FROM Newwar
WHERE NAME1='" & cboRecordChoice.Text & "';"
`grab the record and populate the data
`only grab the fields we need for this label from the database
Err = 0
On Error Resume Next
Set myRecordset = myDatabase.OpenRecordset(sqlStatement)
If Err <> 0 Then
    MsgBox "SQL Error #" & Err & " SQL = " & sqlStatement, vbInformation, "SQL Error"
    Exit Sub
End If
`populate the label fields with the retrieved data
For i = 0 To ClientX1.FieldCount - 1
    thisFieldName = ClientX1.FieldName(i)
    ClientX1.SetData thisFieldName, myRecordset.Fields(thisFieldName)
Next i
myRecordset.Close
pickRandomRecord
myDatabase.Close
`print the label
ClientX1.PrintJob
```

This is a good way to retrieve only the data needed by the label. It also allows your program to be field name independent. New labels can be designed without having to change the Front End System. Using this technique, you no longer have to hard code label information.

**Note:** The key to making this process work is to use actual table and field names in your .LST file that correspond to your database. If a field or a table is added or deleted, you must revise your .LST file to reflect the change. This example only applies to the .NET and ActiveX controls.

An interface is what allows a Front End Program to communicate with the Software Print Server. The Front End makes decisions as to what labels to print based on various criteria. Once this information is known, it is communicated through the LPS Interface to the Back End, where it is processed.

The LPS includes the following interfaces:

- The **File Interface** allows Front End applications to make requests to the LPS via a file drop to a shared network drive. LPS detects this request and responds by printing the label. Any program in any language can use this interface.
- The **TCP/IP Socket Interface** allows Front End applications to have bi-directional communications directly with the LPS through a socket without the need for file transfer, or shared drives. Many aspects of the LPS configuration are available to the front end program through the socket, such as Label List, Field List, and Device Configs. Device status and EPC data pass back are also available in this interface. This interface is for advanced programmers only and is only available in the LPS Premier product. This interface is usually used by UNIX programmers, but can be used by C++ or C# programmers as well.
- The **Direct Socket interface** is similar to the TCP/IP interface mentioned previously. The difference is it is much easier to code to because it is unidirectional. Print Requests are assembled in an XML file and sent through the socket. No status is available other than the fact that if you can't open the socket, the LPS is probably not running. This interface is usually used by UNIX programmers, but can be used by C++ or C# programmers as well.
- The **ActiveX interface** and the .NET interface allow programming languages to send requests to the LPS from anywhere on the LAN, WAN, or Internet. Under the covers, these tools actually use a socket connection for speed and reliability. These tools are available to Windows applications.
- The **On Demand Print Client** and the **Web Client** allow label requests to be made from any client computer on the network, or across the Internet by prompting the operator for keyboard or database key information. No programming is required to use these client programs.

### Related Information

For detailed information on the On Demand Print Client and the Web Client, refer to *Clients* and *Internet Printing* in this guide.

## The File Interface

The File Interface allows Front End applications to make requests to the Software Print Server via a file drop to a shared network drive. LPS detects this request and responds by printing the label. Simply put, the front end will create a "drop" file and write it to the scan directory. Any program on any platform can use this interface and it is the most straightforward interface to implement.

**WARNING:** Do not write data to a file that is in the WDdrop or other scan directory. Data should be written into a file in a non-scan directory and then the file moved to the WDdrop or other scan directory. Adding data to a file in a scan directory may result in a job being stranded in the Pending (olebp) directory without a corresponding Software Print Job (.lpj) file.

When you are first setting up the LPS, you will use the File Interface to test and verify that the system is working properly. You may then use this interface for production or choose another interface that better suits your needs.

Simply set the scan directory as noted in the configuration section, create a PAS or CSV file with the appropriate data and copy the resulting file to the scan directory. Each line in the file must be terminated by a <CR><LF>. Make sure you do a test print first to verify that the printer and label are functioning properly. Once you have performed these tests, you may begin dropping requests to the scan directory from your own application.

## ActiveX / .NET Interface

Programmers developing their own Windows applications in 32-bit languages supporting ActiveX and/or .NET Controls can easily interface their own applications with the LPS. These controls connect to the LPS using a TCP/IP socket and effectively abstract any knowledge of the printer languages from your application. This interface uses properties to set the information needed for the label request and therefore does not follow the PAS, CSV, or XML data structures.

Software has the following controls

- ActiveX Client
- .NET
- Internet ActiveX.

## Software Connectors

The Connector was developed in response to customer needs for easier connectivity between their UNIX systems and the LPS. Oracle applications can be enabled for RFID and barcode printing by installing the Connector with no programming required. Software offers the following connectors.

### Software Connector for SAP Applications

The Software Connector for SAP Applications supports SAP label printing via:

- External Output Management System (BC/XOM XMI) Call
- RFC Listening (Call Function, Destination)

### Software Connector for Oracle

The Software Connector for Oracle serves as a high-speed connectivity bridge between Oracle-based Applications and the Software Print Server.



## Software Universal Connector

The Software Universal Connector serves as a high-speed connectivity bridge between diverse corporate systems and third-party software applications and the Software Print Server

### Related Information

For more information on Software Connectors, refer to the Connector User's Guides on the [software.com](http://software.com) Web site.

## Direct Socket Interface

The Software Print Server (LPS) can be configured to accept direct socket connections that do not require the use of Software's messaging protocol. This connection is a one-way communication that can be easier to program to than the bidirectional TCP/IP interface mentioned previously. The client connects, sends data and closes the connection. There is no response from the LPS. This interface is a quick and easy solution for sending XML files to the LPS through a socket connection on UNIX systems without the need for shared drives or connectivity software.

## Configure the Direct Socket Interface

The Direct Socket Interface (DSI) is off by default for security reasons. If you wish to enable it, follow these steps.

1. Verify that the default scan path is configured to scan for XML files.
  - a. Launch the **Software Print Server Configuration Utility**.
  - b. Highlight the default scan path (the first path listed under the **General** tab) and click **Edit**.
  - c. In the **Scan For** section of the **Directory Setup** dialog, select **XML Files**.
  - d. Click **OK** to accept the settings in both the **Directory Setup** dialog and the **Software Print Server Configuration Utility**.
2. Launch the DSISConfig.exe application in the Software Labeling directory.
3. On the LPS Direct Socket Interface Configuration dialog, select Enable Direct Socket Interface.
4. Leave the Listening Port set to 2813.
5. Click **OK** to accept the settings.
6. If the LPS is running, it must be restarted for the changes to take effect.

**Note:** The Listening Port should remain set to 2813 unless the network or a firewall blocks communication on that port, in which case it may be changed to any other value within the valid port range.

## Integrating with the DSI to print XML files

1. Write a client application that opens a socket connection to the LPS using a call to a Berkeley compliant connect() or a WSASocket() for Win32.

2. Once the connection is established, send a properly formatted XML print job request to the LPS through the socket connection.
3. Close the socket connection.

**Note:** The XML data must be sent all at once. Only one XML request can be sent per connection but it can be a stacked request. Stacked requests for multiple printers are acceptable. The socket connection **MUST** be closed immediately after the XML data is sent.

## Direct Socket Interface Demo Application: DSISend

To further demonstrate how to integrate with the Direct Socket Interface, please see the demo application DSISend. It is written in J# and can be viewed and built in Microsoft Visual Studio. Use any other editor if you simply want to view the code. DSISend is located in the \Sample Programs\Direct Socket Interface\J# Sample\DSISend directory under Loftware Labeling:.

## TCP/IP Socket Interface

Loftware's TCP/IP Socket Interface is a synchronous interface to the LPS that allows submission of a print job, acknowledgement of the print job, and any delays or errors related to the print job. It is much more informative than file-transfer or direct socket solutions, and allows you more control in the customization process. Due to the synchronous nature of this integration method, no other jobs may be submitted until a response is received back for a particular request. The TCP/IP Socket provides for:

- Seamless integration into the LPS.
- No File System Connectivity to host application is required.
- Faster request processing.
- Bi-directional communication between the LPS and the Front-End, meaning, Status Update Responses are sent upon the completion, failure, or delay (due to printer error) of a job request. Information about the labels, fields, and configured printers is also available to your program through this interface.
- RFID EPC data can be passed back to the calling program through this interface.
- The Loftware Connector uses the TCP/IP socket to seamlessly bridge UNIX applications to the LPS. All Loftware printing clients including the ActiveX and .NET controls also use the socket interface.
- This interface is for advanced programmers only. For more basic socket integration, refer to the Direct Socket Interface (DSI) described previously.

The TCP/IP Socket was developed for those computing environments that are unable to use the ActiveX or the File Interfaces. For example, if your application resides on a UNIX or AS400 Platform on your network, but you would rather not use the FTP or shared directories approach, you could optionally communicate directly with the LPS through its socket interface. The LPS ships with a sample program, written in C, that demonstrates how to do this. This program has been compiled and successfully tested on Windows and LINUX platforms. The program is included to give insight into how to use this interface's complex protocol.

We strongly suggest testing your labeling system first by manually creating files for the LPS as described in the *Files Interface* section. Once you have validated printer connections and your labels are printing as expected, switch to using the TCP/IP Socket Interface.

This page intentionally left blank

---

## About the Software Print Server Client Applications

---

Software has developed several client applications in order to maximize the usefulness of the Software Print Server (LPS). Client means that these applications run across the network with a small footprint and interact with the LPS to get their information. Each of the client applications has its own purpose.

### Notification Agent

The Notification Agent is a service that runs on a workstation or on the same computer as the LPS. E-mail and Net Send notifications of printer errors and system status can be set up for multiple operators. Notifications are filtered by server, printer, and error type; therefore, only information of interest is sent.

### LPS Status Client

Allows the viewing of LPS printing activity from anywhere on the network. For individual printers, a tree view of all printers shows queued jobs, jobs with error conditions, and printer status. Any number of LPS servers can be viewed.

For Print Groups, the client allows you to select the print group you want to view and then displays the printers in the group with all their status information.

### On-Demand Print Client

Provides users of the LPS system the ability to manually select label formats and supply data from the keyboard and/or databases to print labels. This client program has the same look and feel as Software's stand-alone On-Demand Print Mode but has the advantage of being thin.

### LPSSend Client

A sample project used as a tool to demonstrate how to integrate with the TCP/IP Socket Interface of the Software Print Server. This program opens a socket connection to the LPS, sends a print job, receives status of the job from the LPS, and then disconnects. The source code for this utility is included and is available for your use.

### ActiveX Client Control

Programmers developing in 32-bit languages supporting ActiveX Controls can easily interface their own applications with the LPS. Use this control when your application needs to print barcode labels or RFID smart labels.

## Internet ActiveX Control

Internet ActiveX (iX) prints across the Internet to locally selected printers that have been configured as CLIENT DEFINED on the server. It is called the Internet ActiveX because it acts as a client across the Internet to the Software Print Server. Use this control when your application is running in several places, needs to access many printers, and requires a small footprint.

## Software .NET Control

Like the other Software Client Controls, this control has a thin footprint. The Software .NET Control utilizes the Microsoft .NET© framework. This control is designed for use in .NET applications.

### Related Information

For information on the ActiveX Client Control see the *ActiveX Client Control* section of the this guide.  
For information on the Internet ActiveX Control, refer to the *Internet ActiveX Control* section of this guide.  
For information on the .NET Control, refer to the *Software .NET Control* section of this guide.  
For examples of the LPSSend Program, refer to *The Software Print Server* section of this guide.

## About the Notification Agent

The Notification Agent is a service that can be run on a client workstation or on the same computer as the Software Print Server. E-mail and Net Send notifications of printer errors and LPS system status can be set up for multiple operators. Notifications can be filtered by server, printer, and error type so that only information of interest is sent to the operator. Please note the following before configuring the Notification Agent:

**Note:** The Net Send Messenger Service is not supported on Microsoft Windows 7, Vista, nor Windows Server 2008. Net Send messaging will not work with the Notification Agent on these Operating Systems. Use SMTP to configure notifications on these operating systems.

- The Agent may run on the same computer as the LPS or as a client on a separate computer. We recommend running as a client so that you can receive a notification if there is a hardware failure on the LPS server and your printing system is down. If the Agent were on the same computer as the LPS, and there were a hardware failure, the Notification Agent would go down as well, and notifications would not be transmitted.
- Although you can run the Agent on several computers on your network, we recommend that you use just one. Its ability to filter specific information to specific operators eliminates the need of using more than one. Running more than one could potentially cause duplicate notifications, or, in the worst case, unpredictable behavior.
- The Notification Agent is only available in the Software Print Server Premier Edition.

## Configuring the Notification Agent

Use the LPS Agent Configuration Utility for starting, stopping, and configuring the Notification Agent. Use the following techniques to launch the LPS Agent Configuration Utility.

On a Client computer (Recommended)

- Open Notification Agent Configuration from the **Start** menu under Software Labeling.

On the Software Print Server server

- Open the **Software Print Server Configuration Utility** (LPS Configuration) from the **Start** menu under Software Labeling.

**Note:** You cannot change the format of the Software Notification Agent log file using the Software Print Server Configuration Utility when the Notification Agent is installed on the same computer as the Software Print Server. When the Notification Agent is installed on a separate computer, you can configure logging independent of the Software Print Server using the Logging tab of the LPS Agent Configuration Utility.

### Notifications Tab

**Note:** The **Notifications** tab is disabled if you do not have the Premier Edition of the Software Print Server (LPS).

From the **Notifications** tab, supply the Notification Agent with the following.

- A list of servers to be monitored
- A list of operators that are to receive notifications
- Email address and/or network computer name for Net Send

**Note:** The Net Send Messenger Service is not supported on Microsoft Windows 7, Vista, nor Windows Server 2008. Net Send messaging will not work with the Notification Agent on these Operating Systems. Use SMTP to configure notifications on these operating systems.

- SMTP settings

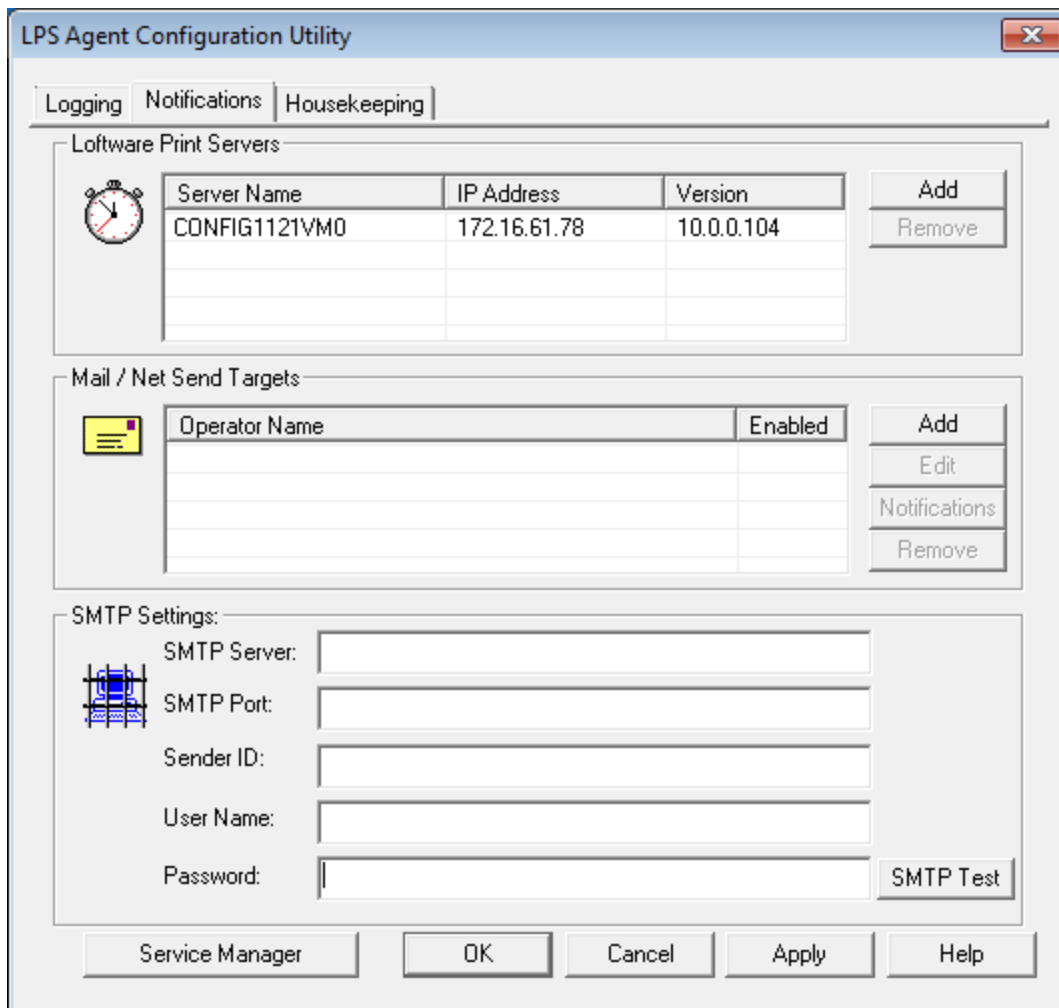


Figure 7.1: Notification Agent Configuration Utility

## Software Print Servers

If you have an LPS running on this server, it is automatically displayed in the server list. If your LPS is on a different server, or if you want to monitor more than one server, click **Add**.

### Add a Server

The following figure displays the **Add LPS Server** window. The server selection list at the top is auto-populated with all the LPS servers that are installed and running on your local subnet. Select the servers in the **Server Selection List** that you want to add (you can pick more than one), and click **Add**. If your server is on a different subnet or is not listed here, you may type in its IP address and click **Ping**. If the ping is successful, the **Name** and **Version** appear in the **Specify Server** section. If an error message is received, make sure that the IP address that you entered is correct and that the LPS at that address is running. Some LPS servers on your network may not show up in this list if they have been configured to not respond to a broadcast. All servers respond to the broadcast by default.



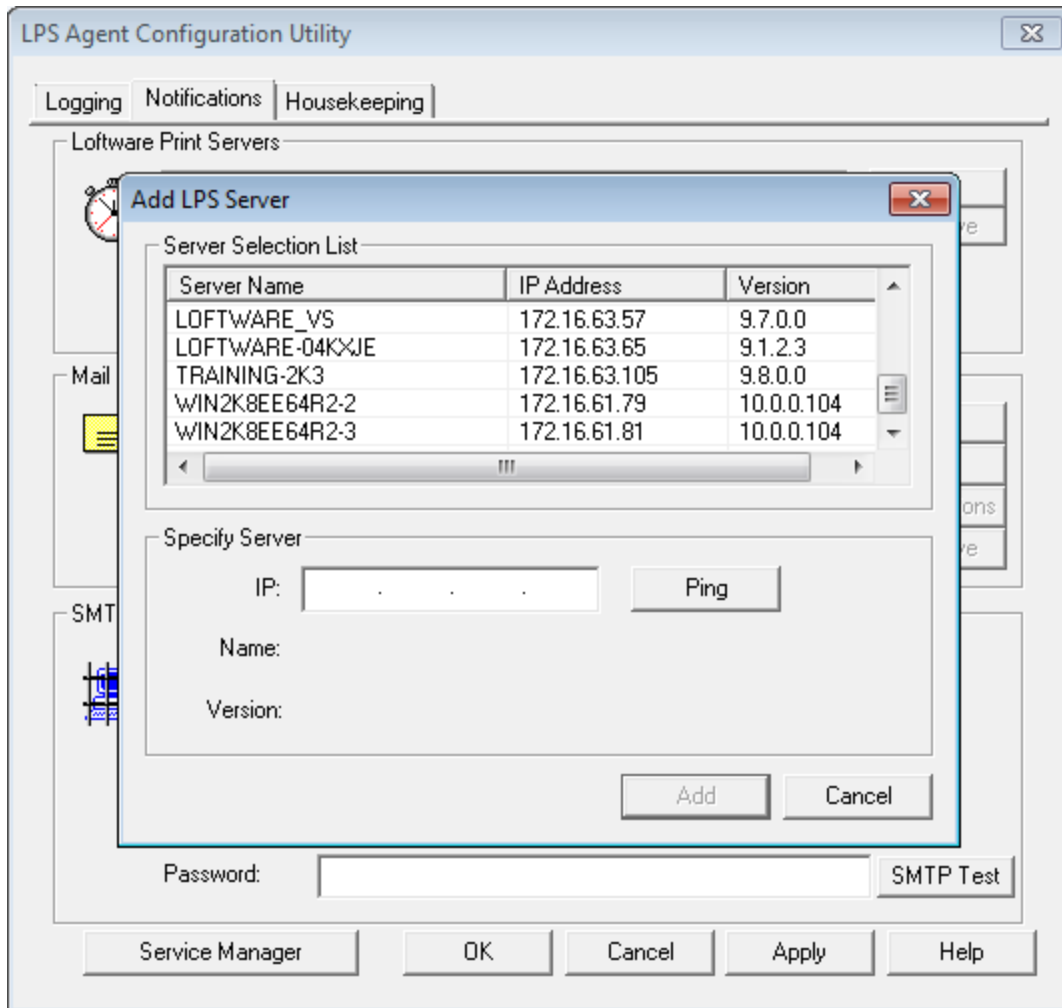


Figure 7.2: Add Server dialog box

## Mail/Net Send Targets

**Note:** The Net Send Messenger Service is not supported on Microsoft Windows 7, Vista, nor Windows Server 2008. Net Send messaging will not work with the Notification Agent on these Operating Systems. Use SMTP to configure notifications on these operating systems.

This display grid lists all operators that you have defined in the system. Click **Add** to create an operator.

Adding a User (Operator)

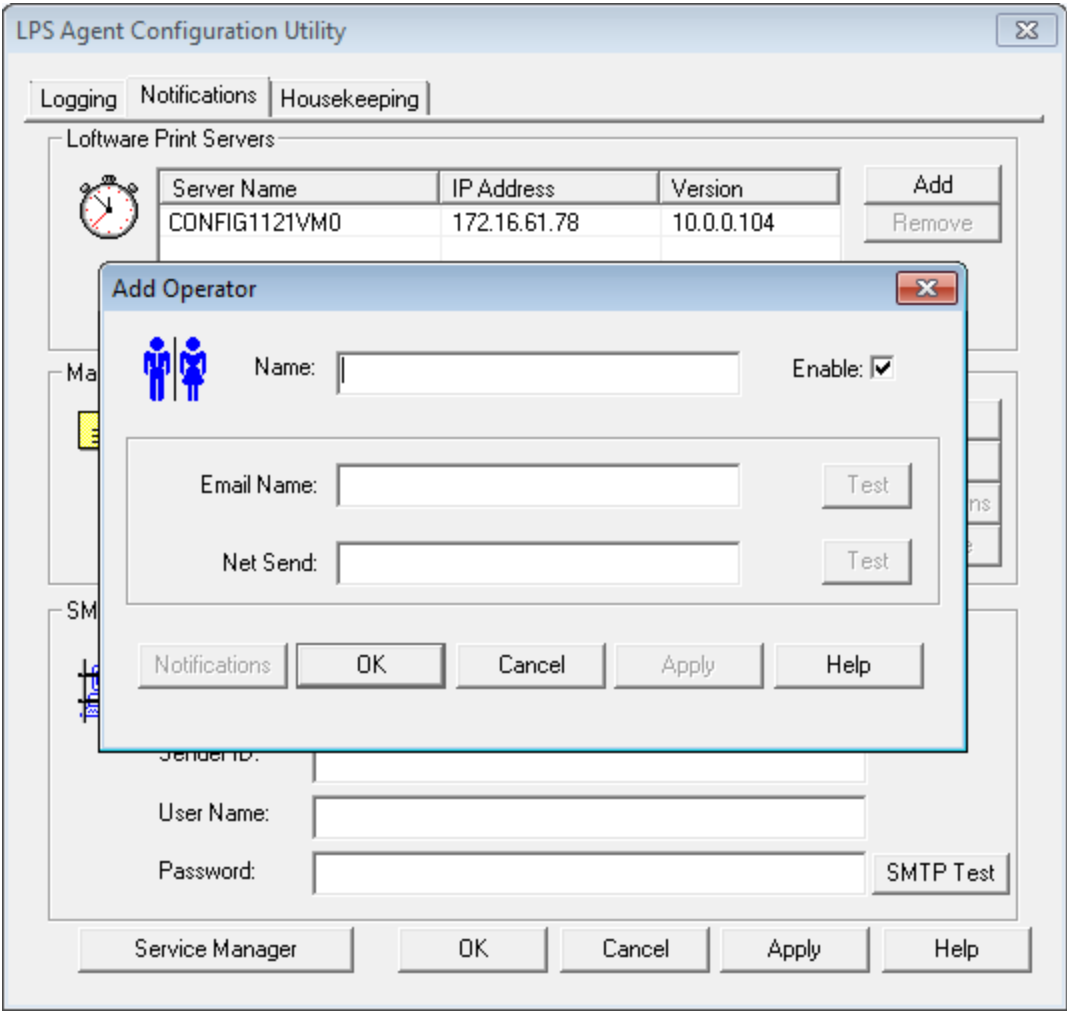


Figure 7.3: Add Operator dialog

Element	Description
Name	The name of the user must be provided here.
Enable	Use this to enable/disable notifications going to an operator. Perhaps they are on vacation and you want to suppress their notifications until they come back.

Element	Description
E-mail Name	<p>An E-mail address or mail system name should be entered here if this user is to receive e-mail notifications. The <b>Test</b> button sends a test e-mail to the address provided. If the test fails:</p> <ul style="list-style-type: none"> <li>■ Be sure that you have selected a valid mail profile on the main screen of the notification tab.</li> <li>■ The address may be wrong; double check it.</li> </ul> <p>If you still have a problem, contact your network administrator.</p> <p><b>Note:</b> Because e-mail networks may be different, Software cannot provide support.</p>
Net Send	<p>Net send sends the notification message across the network to the server name supplied here. The <b>Test</b> button sends a test message to verify that the server name specified is correct. The message appears on the recipient's screen.</p>
Notifications button	<p>All notifications for all devices are sent to this operator by default. The following setup screen allows you to disable all notifications and assign particular ones to this operator.</p>

**Notifications Dialog**

User Profile:  Copy Profile:

Software Print Servers: Configured

☐ Server Shutdown ☐ Mail Notification ☐ Net Send Notification

☐ Device Errors ☐ Mail Notification ☐ Net Send Notification

☒ Enable All Notifications

Figure 7.4: Notifications Setup dialog

When the Notifications setup dialog is first displayed, the servers that you are monitoring are shaded and **Enable All Notifications** is selected. To disable all notifications, clear **Enable All Notifications**, and assign errors for each printer on each server individually for this operator.

Section	Description
Software Print Servers	All LPS servers that you are monitoring are listed on the left side. Click the server for which you wish to assign notifications. You may only select one server at a time.
Server Notifications	Select the <b>Server Shutdown</b> to be notified if the server goes off line. If you have installed the Agent on a separate computer from the LPS, this setting notifies the user if the LPS stops running or the server it is on has a failure or unscheduled shutdown. If the agent is running on the same server as the LPS, notifications are transmitted if the service stops responding, but not if the server itself goes down.
Notify By section	Server shutdowns as well as printer errors and critical failures are transmitted by e-mail, net send, or any combination providing that the address for each has been set up and tested. The <b>Notify By</b> prompt is in both the server and the printer sections because the user may want to be notified in a different way for a server shutdown; perhaps he/she wants to be paged instead. Maybe all printer notifications are sent with e-mail except for Warehouse #2 which is considered critical path. If that printer goes down, you want the operator to be paged. The way these settings work affords you maximum flexibility.
Configured Printers	After selecting a server, the <b>Configured Printers</b> list for the selected server is displayed. There may be a short delay before it shows up. If the list does not show up, the server may be off line or not have any configured printers. Select all printers that you want to monitor on the selected server by ctrl-clicking to select multiple printers.
Device Notifications and Notify By	With the printers selected, choose the types of notifications that you would like to receive by selecting the options. Printer Errors are common errors like Printer not turned on or Printer out of labels. Critical Failures reflect the inability of the server to submit a job because it does not understand it. Some operators may only want to receive critical errors and server shutdown errors. If you have selected more than one printer, these settings apply to all selections. If the option is selected or cleared, (some printers have this setting turned on and some do not) click the option to clear the settings for all selected printers. Click it again to turn the setting on for all selected printers.

**Note:** If your printer supports html status you can use your web browser to get more detailed Printer status.

## SMTP Settings

Field	Description
SMTP Server	The address of the SMTP server to use to send messages

Field	Description
SMTP Port	A valid port. SMTP uses port 25 by default.
Sender ID	The email address of the user from which notification messages should come.
User Name	The username of the sender.
Password	The password of the sender.

### Authentication Types

The Software Notification Agent supports the following authentication types. SMTP servers with no authentication are also supported.

- AUTH LOGIN
- AUTH PLAIN
- AUTH NTLM
- AUTH CRAM-MD5

### Configuring SMTP

- Enter the SMTP Settings for your server.

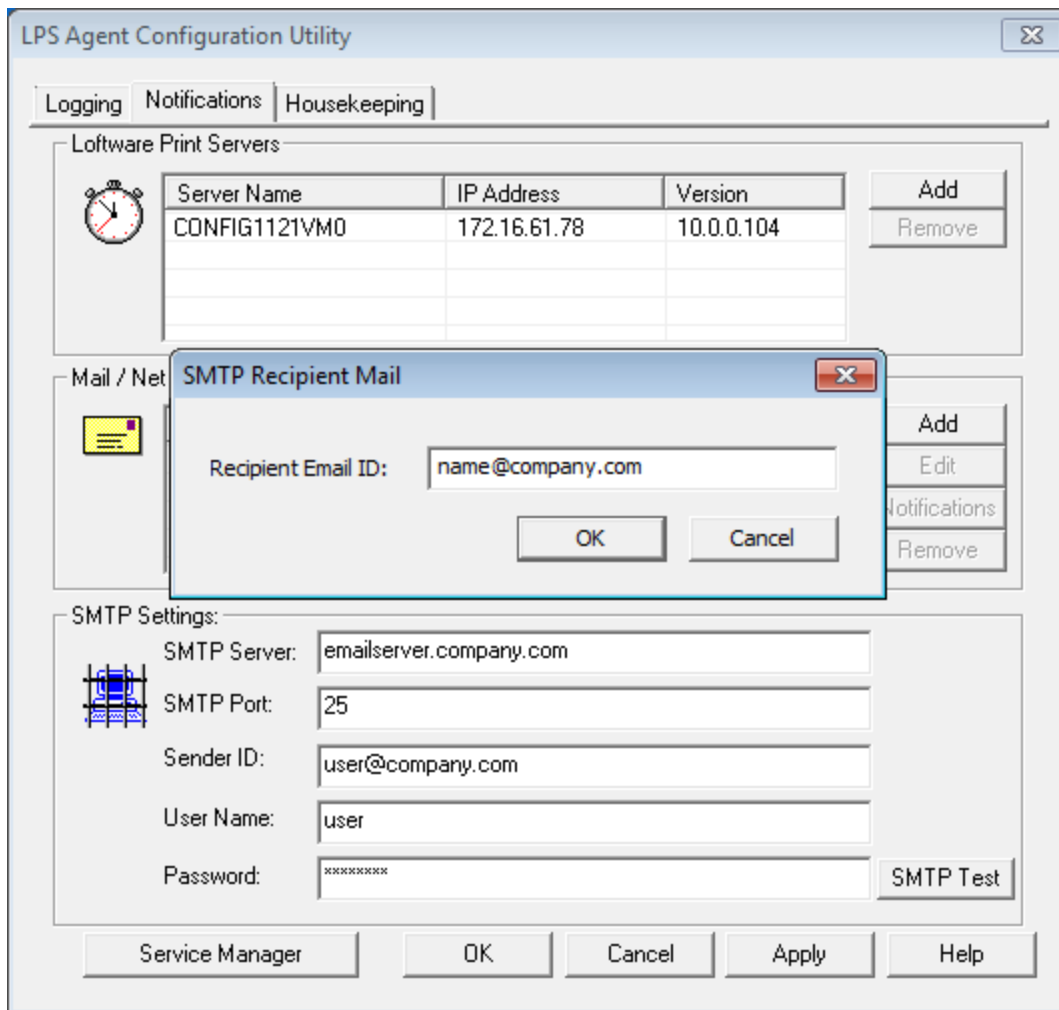


Figure 7.5: LPS Agent Configuration Utility with SMTP Settings and SMTP Recipient Mail

### Testing SMTP

1. Click **SMTP Test**.
2. Enter a valid email address in the **Recipient Email ID** field of the **SMTP Recipient Mail** dialog, and click **OK**.
3. You will receive a Mail Test Passed message, and an email will be sent to the email address you entered.

## Starting the Software Notification Agent

There are two ways to start the Agent. It can be run as a service, or it can be run interactively.

### Service Mode

The Notification Agent is designed to run as a Windows service. A service application is a long-running application that can be configured to run in its own Windows session when the server is started. A

service does not require user intervention, nor does it have a user interface. Services therefore do not interfere with other activities or users on the same computer or server, and are protected from unintentional changes.

You can start the Agent as a service in two different ways. Each method allows you to set the service to Auto Start when the computer starts, so that you do not have to remember to start the service each time your computer is rebooted. The service defaults to manual start. Be sure to run the [LPS Agent Configuration Utility](#) before starting the service for the first time to configure it properly.

**Note:** When you are first configuring the Notification Agent, start it in Interactive mode so you can see error and status information.

## Starting the Notification Agent Service

Do one of the following:

- Click **Service Manager**, select the **Software Notification Agent** and click **Start**.
- Launch the service by clicking **Service Manager** in the **LPS Agent Configuration Utility**. Select the Software Notification Agent as the service to start, and click **Start**.

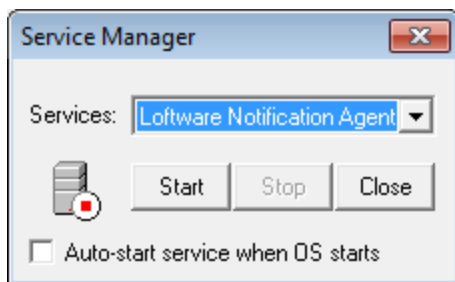


Figure 7.6: Software Service Manager launched from the LPS Agent Configuration Utility

## Interactive Mode

Interactive mode runs the Agent as a normal program and is for system debugging purposes only. There is no protection from the user shutting it down, in which case notifications would stop. It is generally best to use the Agent in service mode when in production.

The Start menu is used to run the Notification Agent interactively. When running in interactive mode, its status window allows you to monitor what it is doing at any given moment.

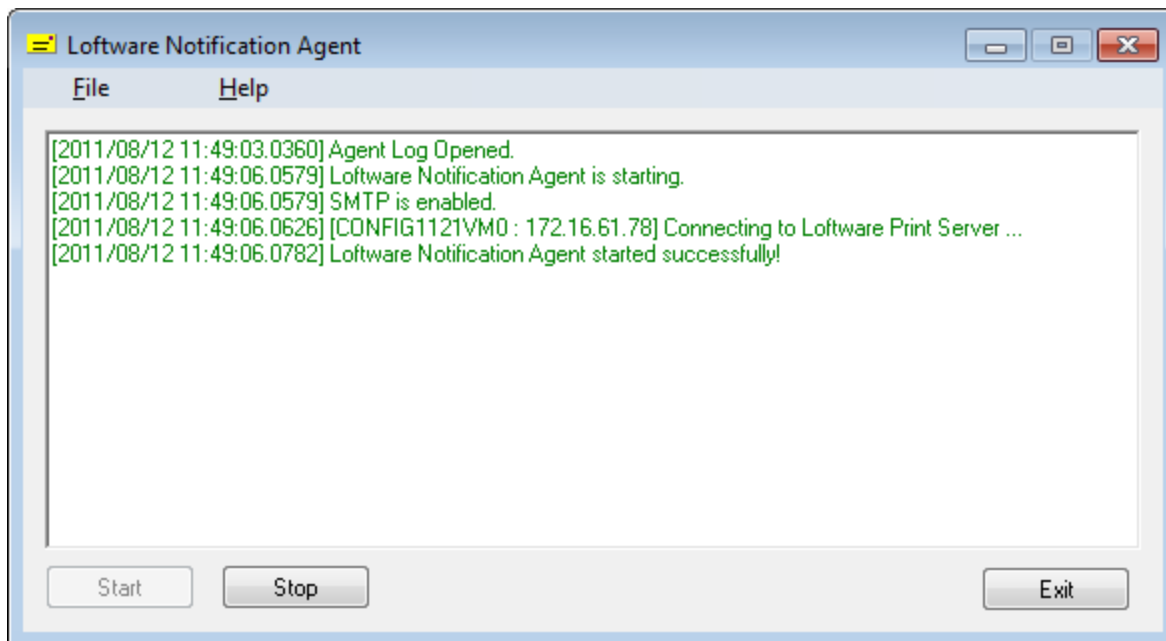


Figure 7.7: Software Notification Agent running interactively

**Note:** If you are running the Notification Agent as a service, you must stop the service before invoking it interactively.

- Use the LPS Configuration Utility to configure the Notification Agent as described in the previous section.
- Start the Notification Agent from the **Start** menu under Software Labeling.
- The Agent can only be started in interactive mode by using **Start**.

## Testing and Troubleshooting the Notification Agent

### Testing the Agent

After setting up the notifications for one or more operators, some simple tests should be done to verify that the Agent is functioning properly. Set yourself up to be e-mailed or net send for all shutdowns, printer errors, and critical errors for all printers (default). Make sure that the LPS and the Agent are started and try one or more of the following to cause a notification transmission:

**Note:** Allow the Agent 30 seconds to connect to the LPS.

1. Shut the LPS down. A notification should go out if you have selected the **Server Shutdown** notification.
2. Create a PAS, CSV, or XML file with an intentional syntax error in it and place it in one of the scan directories. This should generate a critical failure notification.
3. Try turning off printers or unplugging their cables. This should generate a printer error notification when the next print request is made.



4. Some printers generate an error if they are out of labels/or ribbon; others do not. It is instructive to see whether notifications go out under these conditions for your particular printer.

**Note:** Never shut printers off during media error conditions, as the buffer may contain unprinted labels.

#### **Related Information**

For information on printer error messages, refer to the *Device Connections* and *Printers and Labels* sections of the *Loftware Label Manager User's Guide*.

## **On-Demand Print Client**

**Note:** The On-Demand Print Client application is licensed by seats, not concurrency. This means that the first workstations to log in are recorded as a client license taken by the LPS. Once the maximum number of licenses is used, no other On-Demand client modules are able to execute, even if one logs out. If you want different workstations to be able to use client modules, you must stop the LPS Service and then restart it. This clears the license list kept by the LPS. For complete information on licensing for all Loftware Products, refer to the Loftware Web site.

The Client On-Demand Print Module provides users of the LPS system the ability to manually select labels and supply data from the keyboard and/or databases to print labels. The Web Client module has the same functionality but can be used to print labels from a remote location over the Internet.

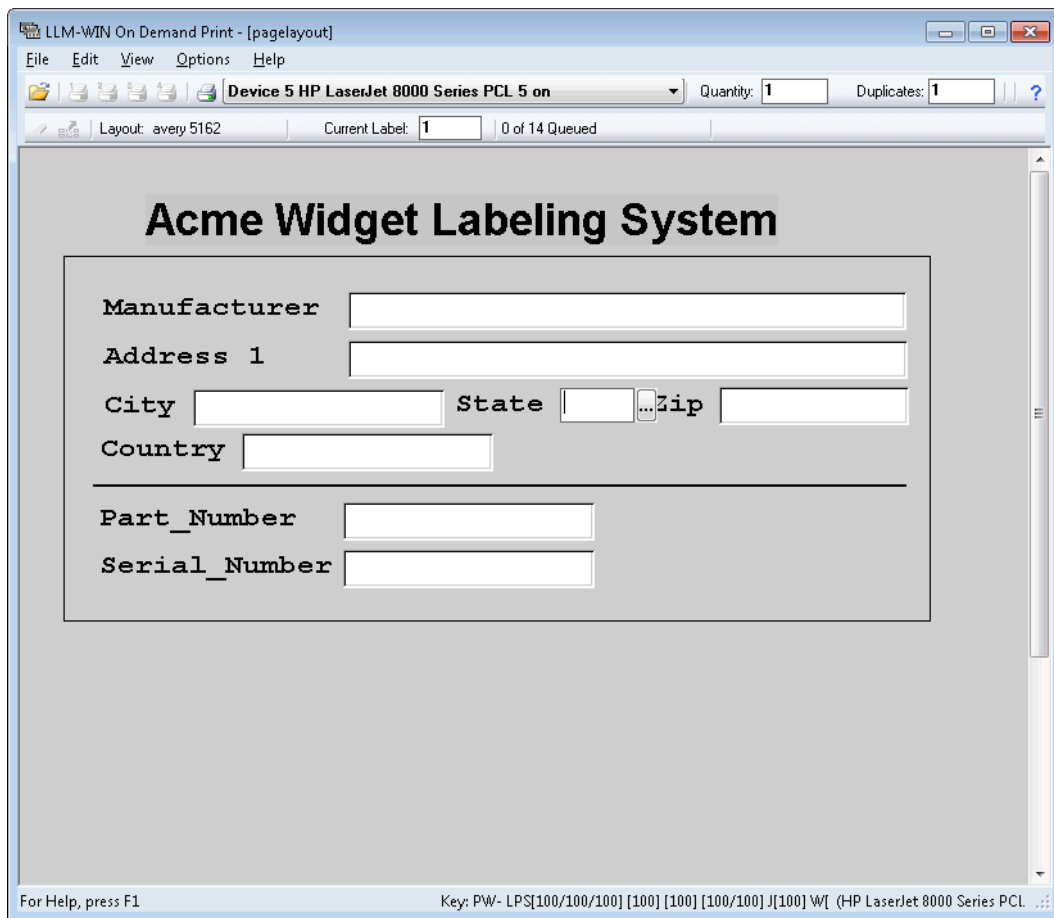


Figure 7.8: Example On Demand Print Client Screen

## On Demand Print Client vs. On-Demand Print Mode

The On Demand Print Client has the same basic functionality as On-Demand Print with the following important differences.

- No hardware license key is required on the Client computer. The one and only license key is located on the same computer on which the LPS is installed. This key tracks how many printers and clients are allowed to use the system.
- The Client requires that the LPS be installed somewhere on the LAN or WAN.
- Although you may have several workstations using the Client, the Software software is only installed on one computer and is therefore far easier to maintain.

If you use one or two print-only stations, it is probably more efficient to use ON-Demand Print within Software Label Manager. If you have three or more print stations, consider using the Client in conjunction with the LPS. Also, consider using the Client if you are already using the LPS System for other purposes and the Client needs to access the same labels and printers already being used by the LPS.

**Note:** The Client only works if you have the LPS configured.

### Related Information

For information on the On-Demand Print mode, refer to the On Demand Print section of the *Software Label Manager User's Guide*.

For information on installing and configuring the LPS, refer to the *Software Print Server and Software Label Manager Installation Guide*.

## Preparing to Use the On-Demand Print Client

**Note:** This topic assumes that you have already installed the client.

1. Connect your printers, and configure them.
2. Design your labels, making sure that you customize your operator input screen.
3. Test print your labels from Software Label Manager before trying to use the On-Demand Print Client.
4. Print your labels using the On-Demand Client.

## Troubleshooting

If your labels do not print.

- Make sure that you are able to test print your labels using the Client from the server before trying to use the Client on a workstation.
- Stop and start the Software Print Server and try again.
- Use the llmwcInt.ini configuration file to make any customizations for advanced users.

## How the On Demand Print Client Works

When the ODP Client is launched, it sends a broadcast looking for all LPS servers on your network (local subnet). When the client receives the response, it connects to the server automatically. If no servers are found, the launch fails. If more than one server is found, an arbitration dialog box is displayed prompting you to choose the server you wish to use. If your LPS server is on a different subnet, or if you wish to have access to multiple LPS servers, create an llmwcInt.ini file. See [The llmwcInt.ini File](#) section for more information .

**Note:** If, for security reasons, you want to suppress a server on the arbitration list, call Software Technical Support at 603-766-3630 (follow the phone prompt for Tech Support) for assistance with this option.

The Client needs access to the following LPS subdirectories: The first, called "labels" is where the .lwl file is obtained in order to prompt the operator. The second, called "layouts" is the directory where the layout files are stored. The third, called "wddrop" is the default LPS scan directory where the print requests are directed. If images and serial numbers are being used, access is required to these directories as well.

By default, a global share is created during the install of the LPS called **loftware\$**. All directories mentioned previously can be found under this share. If you have changed the location of any of these directories, the client must have privileges granted to it in the form of a share. See Knowledge Base article 2009213 on the Software Web site for more information on share points.

**Note:** You may choose to limit the **loftware\$** share to users of the client programs. By default, it is a global share, which can be dangerous. Use Windows Explorer to accomplish this.

## The llmwcInt.ini File

In most cases, this file is not utilized. The directories referred to in the previous section are the defaults, and thus a llmwcInt.ini file does not need to be created unless the client computers are located on a separate subnet from the LPS Server.

The .ini file should be named llmwcInt.ini and should be in the same directory as the Client. When LPS is installed, it installs a sample file called llmwcInt.sav to the same directory as the LPS. Renaming this file with an .ini extension gives you a head start in creating the file.

The following is a example of what a llmwcInt.ini file looks like if you have LPS installed on two servers.

```
[Config]
;ForceSelectPrinter=1
;iniRedir = \\server1\Software Labeling

[Receiving1]
Name=JANAA
Alias=Jana Computer
Address=172.16.0.8

[DemoRoom]
Name=TRAINING
Alias=Training Room Server
Address=172.16.0.9
```

Section/Command	Description
[Config]	General Client Configuration settings section.
ForceSelectPrinter	Forces the client to choose a printer based on the selected label to prevent accidental misrouting of print jobs.
IniRedir	If you wish to locate the client executable on individual computers to reduce load times across your network, you can create an llmwcInt.ini file on each computer with the <b>iniRedir</b> section pointing back to a full .ini file that is centralized. This way, if anything in the file needs to be changed, you still have central control. If you have performed a 'thin install', the llmwcInt.sav file is created on the client computer for you.
[Section]	Section names must be unique and surrounded by [ ] s. Each section name, except config, must represent a unique name of an individual LPS print server.
Name	The computer name on the server.

Section/Command	Description
Alias	The common name for the server.
Address	The IP address of the computer running the LPS. The system automatically receives this address using a broadcast technique if the address entry is not there, unless it is on a different subnet, in which case it is not found.
LabelsPath	The UNC path to the labels directory if different from the default. (ClientX)
LayoutPath	The UNC path to the layout directory if different from the default. (ClientX)
PrinterPath	The UNC path to the LPS directory. (ClientX)
ScanPath	The UNC path to the scan directory if different from the default. (ClientX)

## Operation of the On-Demand Client

1. Launch the Client from a shortcut or **Start** as described previously. The **Software On-Demand Print Client** window is displayed.
2. From the On-Demand Print Client File menu, select **Open**. Choose the label file that you want to print from the Labels directory.
3. Enter label information using parameters as shown in the following figure.

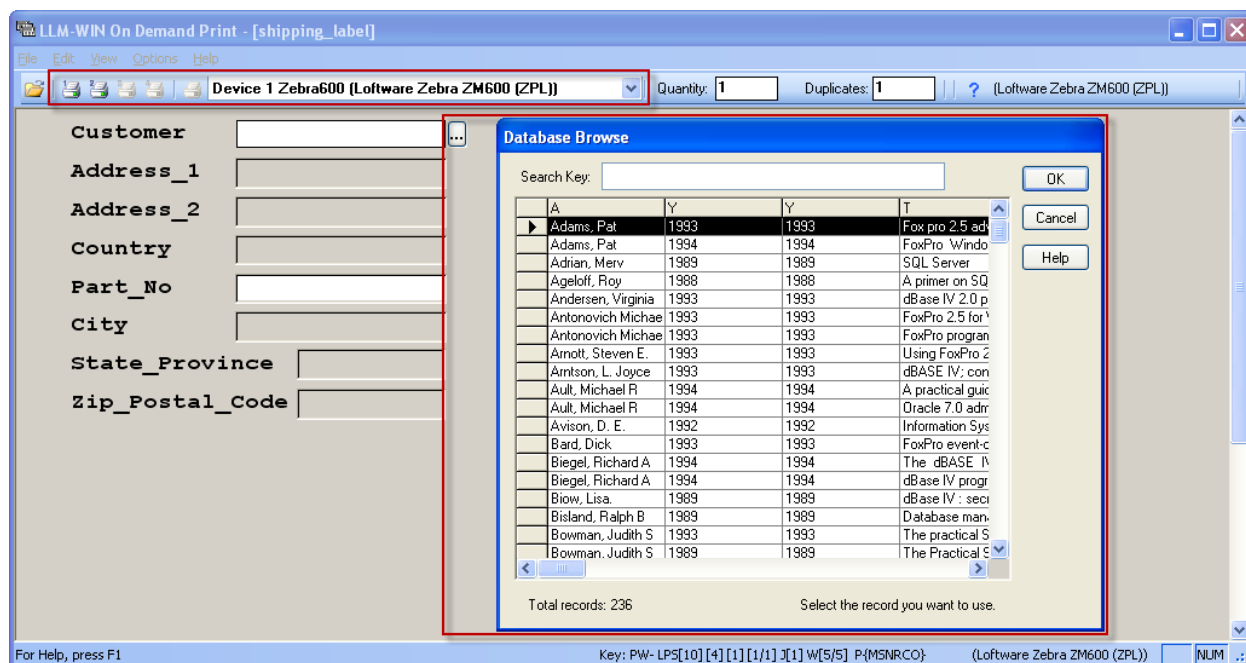


Figure 7.9: Client On-Demand Print Data Entry Screen

## Using the ODP Client with a Database

In the previous figure, certain fields are grayed out except the Database Key field. Use one of the following options to access the database information for your label.

## Browse the database

1. Click the browse button (...). The Database Browse window opens with the first page of records in the database displayed.
2. Type character(s) in the **Search Key** field. When a match is found, it is highlighted.

## Filter the database

1. Enter one or more valid character(s) in the key field and click the browse button (...). The **Browse Filter** dialog opens.

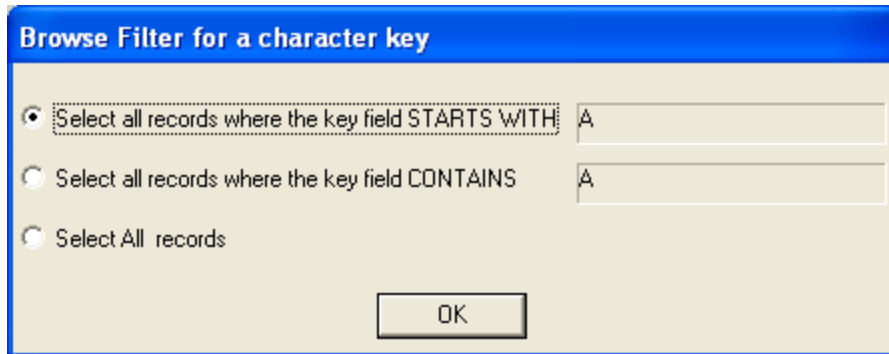


Figure 7.10: Browse Filter

2. Choose one of the filter options. The database is filtered to display only database fields that conform to the filter.

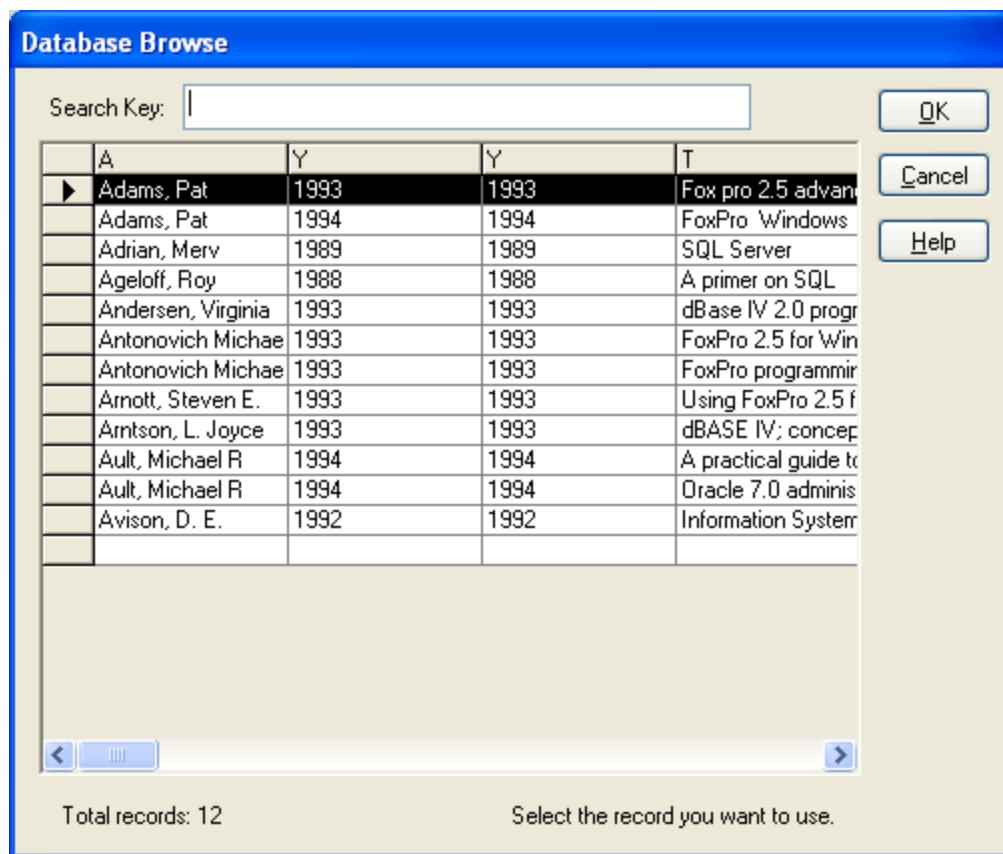


Figure 7.11: Browsing the database using the Search Key

3. Double-click the record you want to open.
4. Print the label(s) by clicking on one of the printer icons, select File | Print from the menu, or press F9 on the keyboard.

## Creating an Icon to Run the Thin On-Demand Print Client

Creating an icon that is attached to a particular label on a desktop provides instant printing capability. When clicked, the ODP Client is opened with the label requested, and the label is printed.

### To create the icon

1. From the Software Labeling directory in Windows Explorer, locate and right-click LLMWCInt.exe.
2. Select **Create Shortcut** from the menu.
3. Click and drag this shortcut to your desktop; close Windows Explorer.
4. Right-click the desktop icon, select **Properties**. This displays the LLMWCInt.exe **Properties** box.
5. Add the name of the label that is to be printed at the end of the Target line after .....exe.

#### Example

"C:\<Program Files Folder>\Software Labeling\LLMWCInt.exe" db\_1.lwl

Where the path to the label is a UNC path to the LPS Server plus the Software share name:

```
"C:\<Program Files Folder>\Software Labeling\LLMWCInt.exe" -L "\\lps-server  
hostname\SOFTWARE$\labels\db_1.lwl"
```

## To specify a printer or quantity

If you want to automatically select a printer on a per user basis, add the `-p` command to the shortcut:

`-pn` (Where `n` is any valid, configured Software Printer number)

The same format is used for the for the default quantity:

`-Dn` (Where `n` is a valid, non-negative default quantity)

**Note:** If the `-p` command is used, and you open a label that is not designed for the printer specified, the `-p` command is ignored.

For example, the client is executed with `-P4` and Printer 4 is an Intermec EasyCoder 3440, the user opens `acme.lwl` which is designed for a Monarch 9830, the default printer of 4 is ignored since the 3440 is not displayed in the printer combo on the client (remember the only printer(s) displayed match the printer(s) the label is designed for in the Client).



## Troubleshooting the On-Demand Print Client

For information on your Client session, select View | Diagnostics from the menu bar.

The Software Client Diagnostics dialog box is displayed.

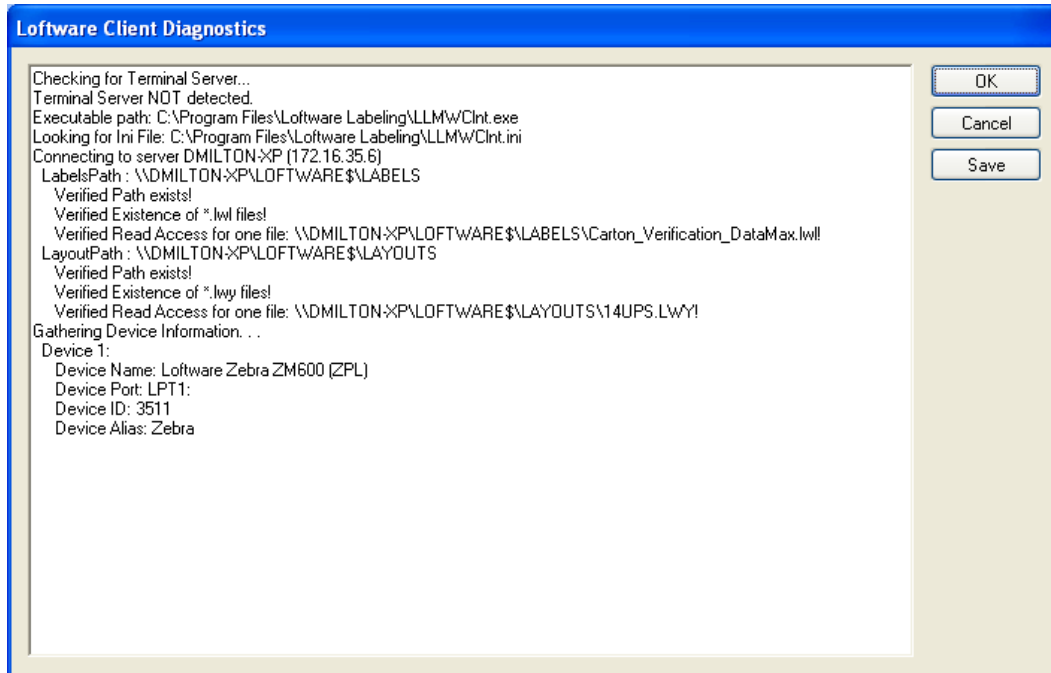


Figure 7.12: Software Client Diagnostics dialog box

The Client Diagnostics box gives you information on the Labels, Layout and Scan Paths, as well as printer information. If you are still having trouble printing using the Client On-Demand Printing module, read the following checklist before calling Technical Support:

- Verify that the LPS is running (service started or scanning in interactive mode).
- Verify that the paths you have chosen for your labels, layout and the scan path are valid. If they are not, manually set the paths in the UNC paths section.
- Make sure that you have configured the right printer for the label.
- Verify that all server information is correct, such as the IP address, etc.
- Make sure that you can still test print the label from the label designer back on the server.

It is possible to configure printers with the Software Print Server running. The LPS monitors printer changes, and when it detects a change, the printer list is updated, and notifications are sent to the connected On-Demand Print Client(s).

Be aware that a problem could develop on the client-side if a printer is deleted on the server side; for example, an ODP Client attempting to print to the deleted printer would receive a display that states “No Printer Configured.” Also, when the list of printers is updated because of a configuration change, if more than one printer of the same make and model printer exists, the default is to the first printer in the list that matches the loaded label.

**Example**

Three Zebra 170XiIII printers are configured on Printers 1, 5 and 7. The user is printing to Printer 5, and the printer configuration is changed. The default printer then becomes Printer 1. This may create a problem for a user who is attempting to print and does not notice that the printer list has changed. The label that was previously printing without any problems seems not to be able to print. This is another example where a Printer Alias can be very helpful, as the client user may be more apt to notice a change in the alias of the printer, as opposed to just a number.

## About the Status Client

The LPS Status Client allows the viewing of LPS printing activity from anywhere on the network. It allows you to get a quick snapshot of printing activity throughout the shop floor. It also allows you to delete pending jobs and reprint jobs on an as needed basis. Tree and Context views are provided to maximize the information that you can obtain.

LPS Status Client facts:

- Any number of LPS servers can be monitored. Jobs can be viewed, resubmitted, or deleted.
- You are limited to viewing one print group per server on a Status Dialog session, but you can quickly select another group from the Group drop down menu.

**Note:** If, for security reasons, you want to suppress a server on the LPS Server list, call Software Technical Support at 603-766-3630 (follow the phone prompt for Tech Support) for assistance with this advanced option.

**Note:** If you do not have the Premier Edition, Status is only run on the LPS server.

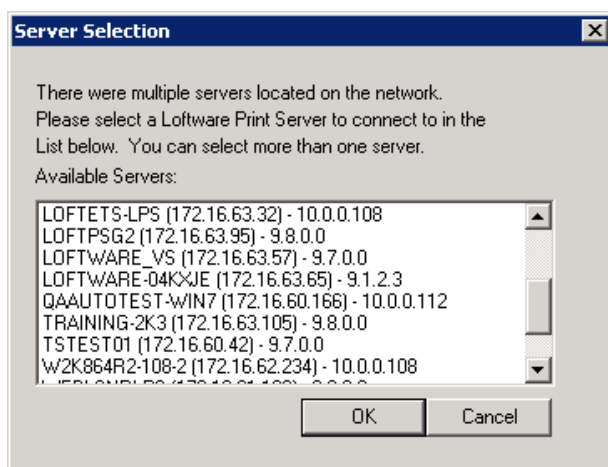
### Related Information

For information on how to launch the status program from either a shortcut or **Start** after performing a thin install, refer to the *Installing and Starting Client Applications* section of this guide.

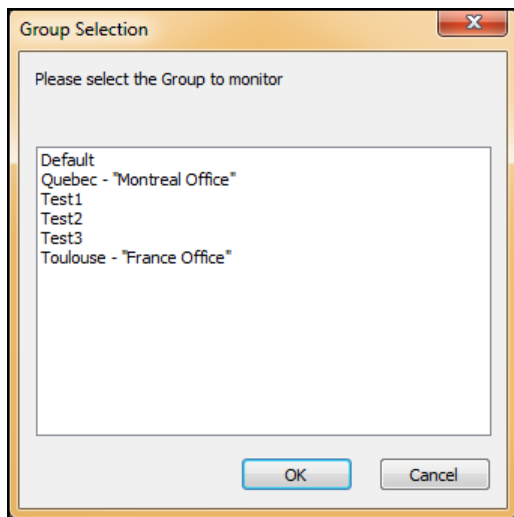
The status client uses much of the same technology as the On-Demand Print Client. To review this information, refer to the *How it Works* and *lmmwclnt.ini* sections of the *On-Demand Client* section in this guide.

## Operating the Status Client

The Status Client application can be launched from a shortcut or the **Start** button. At startup, the Status Client broadcasts to all the LPS servers on your local subnet. If it finds more than one, the following dialog box is displayed. Multiple servers may be selected for monitoring.

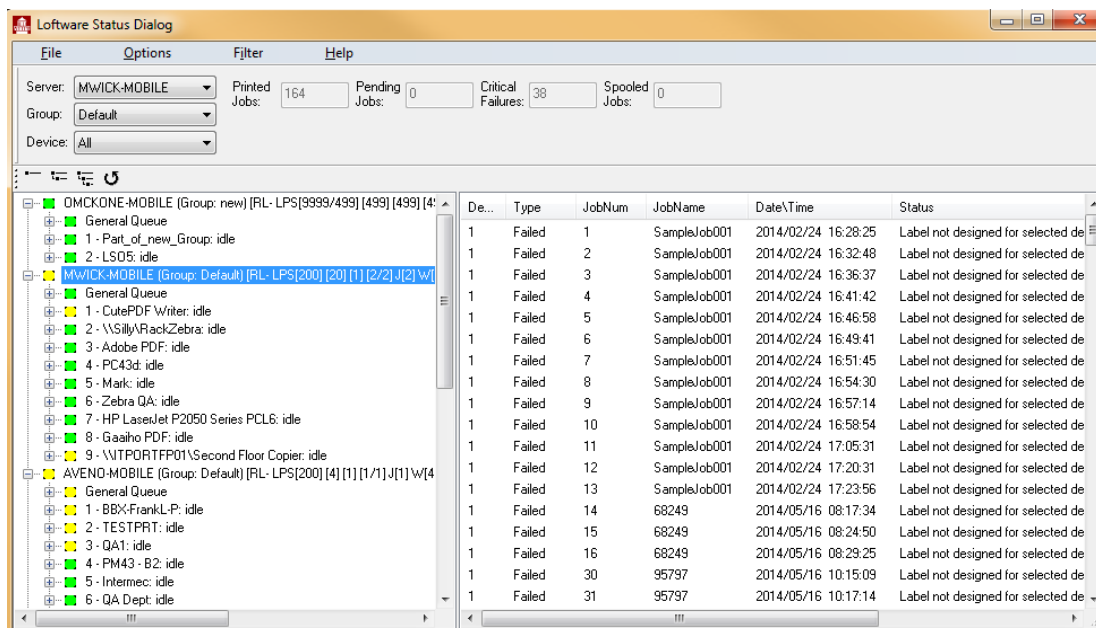


If the server you select is using the Software Label Manager Print Group feature, then you are given a second dialog to choose one Print Group that is defined on the server. The Status Client is limited to viewing one print group per LPS server at a time. However you can select all the groups, one at a time from the Software Status Dialog's Group drop-down menu.



**Note:** With LLM Print Groups, all individual printers are automatically assigned to the Default Group, until they are assigned to a new Print Group. Refer to the *Software Label Manager's User Guide* (online Help or PDF versions) for more information.

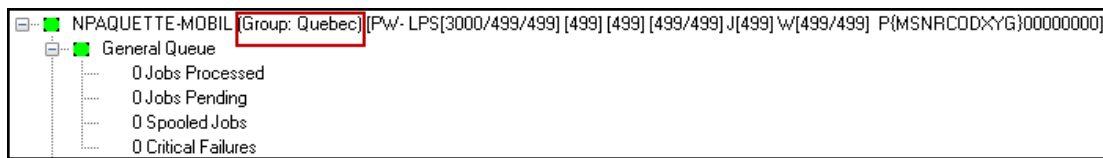
The Status Client dialog displays a server (and group)/printer tree on the left side and a context view on the right side. It is called a context view because its display is dependent upon which branch of the tree is selected.



On each server there is a **General Queue** specified for a print group, either the Software-supplied Default group, as shown above, or a user-defined group. Error conditions are logged when the system cannot figure out which printer is to receive the request.

For example, if you dropped a file that had a syntax error that caused LPS not to be able to parse the destination printer, the error would go into the general queue. This queue is most helpful for reviewing mistakes that you have made during the development process. Once your requests are coming through with no syntax errors, the General Queue should not be needed.

The example below shows a user-defined print group named, Quebec.



## Adding Servers

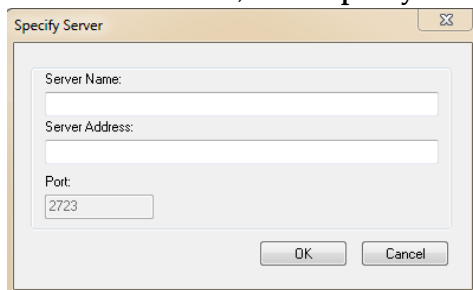
If you already have the Status Dialog running, you can add additional LPSs servers to the Status Dialog display.

**Note:** To add new servers to the Status Dialog you will need to know the Server Name and Server Address. Software recommends you record this information prior to performing these actions.

**Note:** Software support is limited to viewing one print group at a time per LPS server in the Status Dialog.

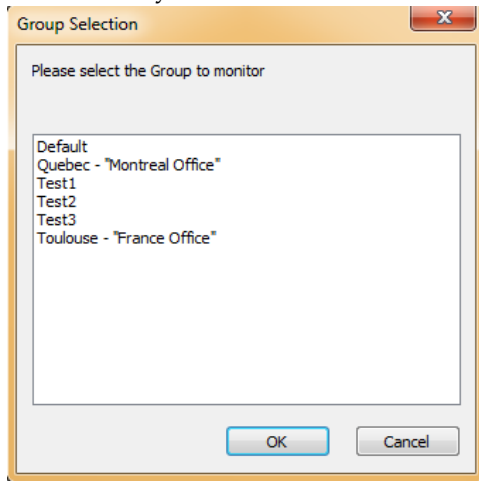
### Adding a New Server

1. From the File menu, select **Specify Server** to display the following dialog:



2. Enter the **Server Name** and **Server Address**.
3. Click **OK** to display all the printers on the Server's Default Group in the Status Dialog.

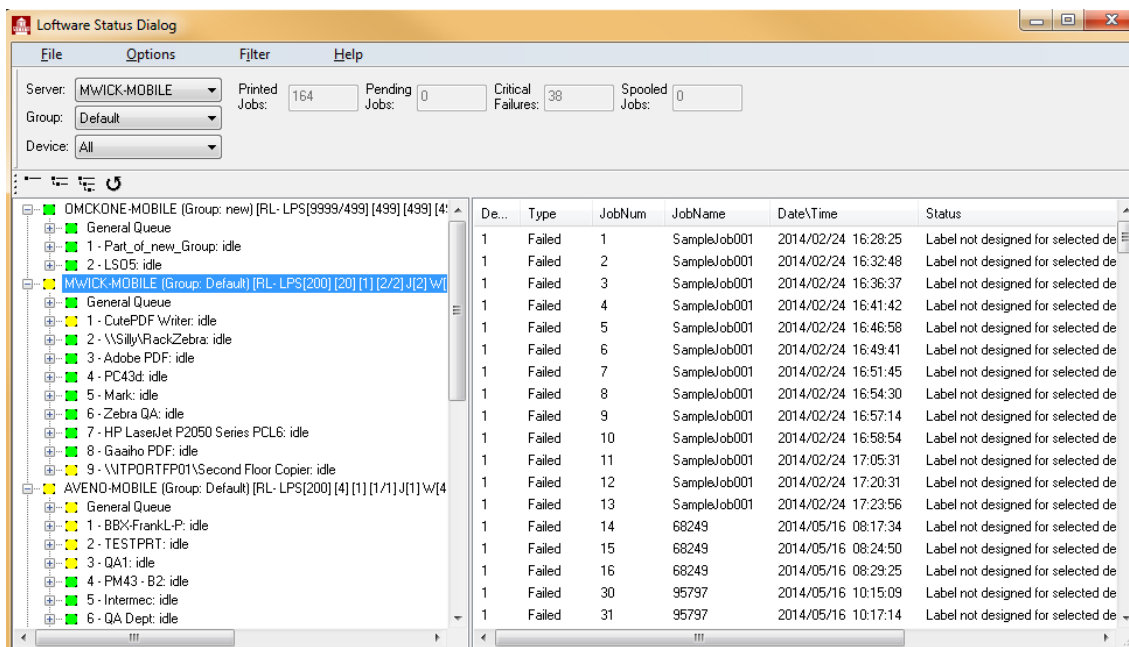
If the server you selected has additional print groups defined, the following dialog will display:



4. Select the one group on the LPS server you want to display in the Status Dialog and click **OK**.

## Understanding the Screen Layout

The Status Client can provide a wealth of information. Hundreds of printers, or printer groups, being driven on multiple servers can be monitored simultaneously.



## Real Time Refreshing

All limbs of the selected (highlighted) branch of the tree are updated 'real time.' There may be a slight delay depending on how busy the server is. You can make all printer status for a particular server

update in real time by selecting the server. The server or servers are selected by default when entering the program. You can select the server you want to monitor from the Server drop-down menu, or by clicking on the server in the left-hand tree display.

## Refreshing at Polling Interval

Any limbs that are not part of the selected branch are updated at the polling interval, which is 30 seconds by default. Shortening the poll interval speeds the screen update but burdens the server more. Software recommends that you leave the poll interval at 30 seconds and use the real time highlighting method described previously.

## Purge Level

Jobs in the context view remain until the next purge interval. See the **Housekeeping Tab** section of the *Software Print Server* section for detailed information on file purging.

## Jobs marked as Printed

When a job in the context view is marked as Printed, the LPS has finished processing this job and it remains on the system until the next purge interval. This type of job can be resent to the printer as described in one of the following examples.

### Jobs marked as Pending

If the printer is in a busy or error state, new jobs that are requested of it are queued up and marked as Pending. Pending jobs can be deleted by right-clicking on the job and choosing Delete from the context menu. Pending job files allow the LPS system to recover from critical errors and are not deleted during the purge process.

Keep the following points in mind when deleting a Pending job:

- Depending on how busy the server is, there can be a considerable pause before the job is removed.
- When you delete a Pending Job, the printer goes into an error state temporarily. Notifications go out if you are using the Agent. Normal printer operations resume after the next print request to that printer.

### Jobs marked as Failed

If the LPS does not understand the request that has been made, it is marked as 'Failed.' This occurs when the request contains a syntax error or invalid information. Failed jobs usually occur when you first start debugging your system. They should become rare or non-existent when your system is fully debugged. You can "right click" on a failed job and choose View File. If you see a syntax error, you may correct it and hit resend.

### Jobs marked as Spooled

These are jobs that have been processed by the LPS and are being sent to the Web Listener for processing. Once the job request has been sent back to the LPS by the Listener, the context view for that job is displayed as "Requested." Once the job has been printed or fails, it moves to the Printed or Failed directory. Spooled jobs can be deleted by right-clicking on the job and choosing Delete from the context

menu.

### **Green icons on tree branches**

A green icon indicates that the printer is accepting jobs and no errors have been detected. For more information, refer to the "Device Status" section of *Loftware Label Manager User's Guide* and the "Printer Error Message" section of the specific printer family in the *Loftware Label Manager User's Guide*.

### **Red icons on tree branches**

A red icon indicates that a printer is in an error state. No new jobs are printed on this printer until the error is resolved. The icon at the server level goes red if any of its printers (branches) have a problem. This does not mean that the server is down; it simply is indicating that one of its printers needs attention.

### **Yellow icons on tree branches**

Indicates that there was a Failed Job on that branch. The reason it is yellow is that although there was a failed job, new jobs that do not have syntax errors are still able to print. Like the red icon, the server icon goes yellow if one of its printers goes yellow.

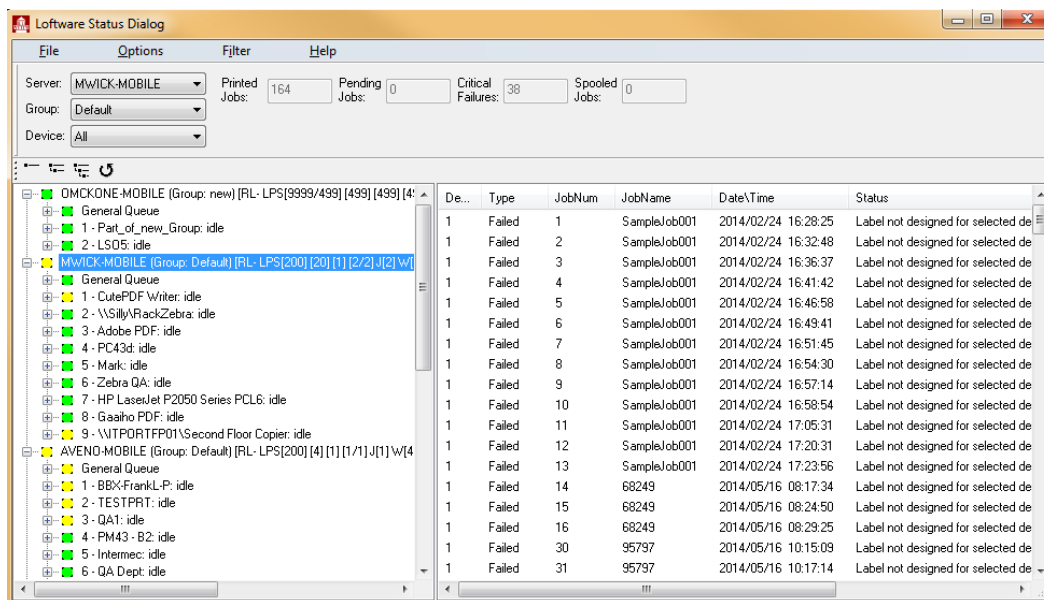


## Status Client Examples

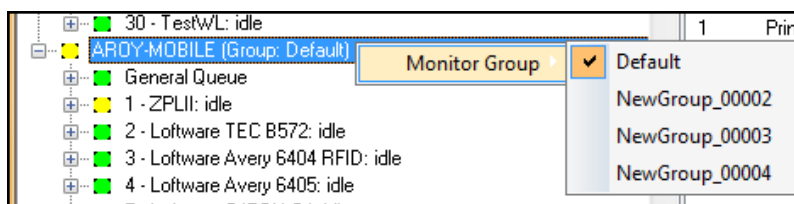
### Viewing Jobs

The figure below shows the status of an LPS server.

- When the server (and/or print group) is selected on the tree, the context view shows all pending and printed jobs for all printers on the selected server in real time.
- If a branch is expanded showing Printed/Pending/Spooled/Failed jobs, the branch refreshes in real time, and all other branches refresh at the polling interval.



- If the server has multiple print groups, you can select which group to view from the Group drop-down menu or by right-clicking on the server in the tree display, and then selecting the print group.



## Context Menus

You can use the Context Menu to view a file, resend jobs, and delete jobs.

The next figure shows another status view of an LPS server:

- The number next to the name is the serial number of the LPS running on this server.
- The count in parenthesis next to the printers reflects the number of pending jobs for that particular printer.
- The context menu shown in this figure is displayed when an item in the context view is right clicked. The choices in this menu are documented in the Menu Choices section.

1	Printed	1	UnicodeLineSpace	2015/11/10 ...	(M5492) Printed
1	Printed	2	UnicodeLineSpace	2015/11/10 ...	(M5492) Printed
1	Printed	3	UnicodeLineSpace	2015/11/10 ...	(M5492) Printed
1	Printed	4	UnicodeLineSpace	2015/11/10 ...	(M5492) Printed
1	Printed	5	UnicodeLineSpace	2015/11/10 ...	(M5492) Printed
1	Printed	6	UnicodeLineSpace	2015/11/10 ...	(M5492) Printed
1	Printed	7	UnicodeLineSpace	2015/11/10 ...	(M5492) Printed

View File  
Resend Job  
Resend Selected  
Delete Job  
Delete Selected  
Delete All

## Monitoring Multiple Servers

The following figure shows the Status dialog connected to multiple servers.

- Three LPS servers are shown in the tree.
- The LPS server **OMCKONE\_MOBILE** (and print group **new**) is marked as green because all the print jobs for all the printers in the group have printed successfully.
- Server **MWICK-MOBILE** (and print group **Default**) is expanded to show its General Queue and 9 devices. The right column displays all the Printed, Pending and Spooled jobs for that server.
- Server **AVENO-MOBILE** (and print group **Default**), can be viewed when selected from the Server drop-down menu or directly selected in the tree panel.

The screenshot shows the 'Software Status Dialog' window. At the top, there are tabs for 'File', 'Options', 'Filter', and 'Help'. Below the tabs, there are dropdown menus for 'Server' (set to 'MWICK-MOBILE'), 'Group' (set to 'Default'), and 'Device' (set to 'All'). To the right of these are counters for 'Printed Jobs' (164), 'Pending Jobs' (0), 'Critical Failures' (38), and 'Spooled Jobs' (0).

The main area is divided into two panes. The left pane shows a tree view of servers and their queues. The right pane shows a table of print jobs.

De...	Type	JobNum	JobName	Date\Time	Status
1	Failed	1	SampleJob001	2014/02/24 16:28:25	Label not designed for selected de
1	Failed	2	SampleJob001	2014/02/24 16:32:48	Label not designed for selected de
1	Failed	3	SampleJob001	2014/02/24 16:36:37	Label not designed for selected de
1	Failed	4	SampleJob001	2014/02/24 16:41:42	Label not designed for selected de
1	Failed	5	SampleJob001	2014/02/24 16:46:58	Label not designed for selected de
1	Failed	6	SampleJob001	2014/02/24 16:49:41	Label not designed for selected de
1	Failed	7	SampleJob001	2014/02/24 16:51:45	Label not designed for selected de
1	Failed	8	SampleJob001	2014/02/24 16:54:30	Label not designed for selected de
1	Failed	9	SampleJob001	2014/02/24 16:57:14	Label not designed for selected de
1	Failed	10	SampleJob001	2014/02/24 16:58:54	Label not designed for selected de
1	Failed	11	SampleJob001	2014/02/24 17:05:31	Label not designed for selected de
1	Failed	12	SampleJob001	2014/02/24 17:20:31	Label not designed for selected de
1	Failed	13	SampleJob001	2014/02/24 17:23:56	Label not designed for selected de
1	Failed	14	68249	2014/05/16 08:17:34	Label not designed for selected de
1	Failed	15	68249	2014/05/16 08:24:50	Label not designed for selected de
1	Failed	16	68249	2014/05/16 08:29:25	Label not designed for selected de
1	Failed	30	95797	2014/05/16 10:15:09	Label not designed for selected de
1	Failed	31	95797	2014/05/16 10:17:14	Label not designed for selected de

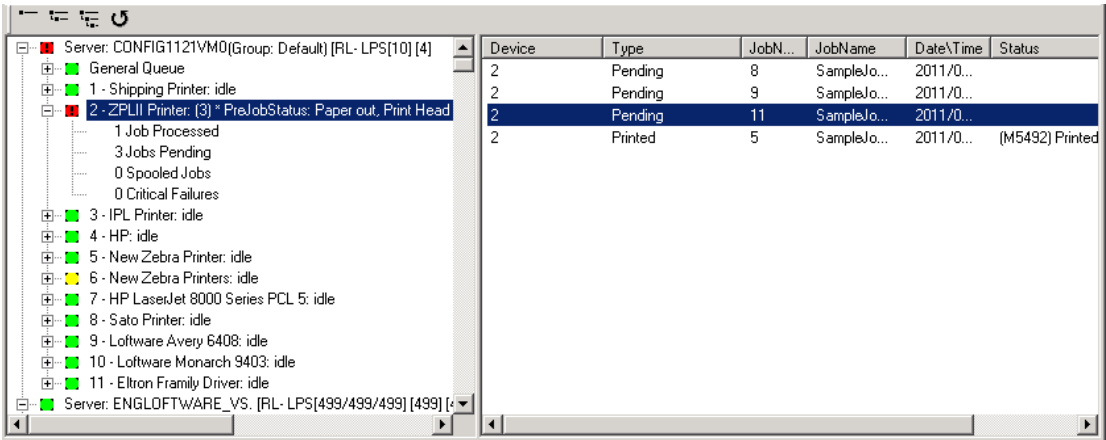
Interpreting Error Conditions

This example shows a server named CONFIG1121VM0 (and print group Default) that has several configured devices.

- A red exclamation point (!) icon to the left of the server name indicates that there is an error on one of its devices.
- The icon to the left of Device 2 (ZPLII Printer) is red because it is the printer with the error.
- Note the error messages after the printer name. These particular messages are available because Detailed Status is enabled for this printer - this option is available for certain Intermec and Zebra printers. See the *Software Label Manager User Guide* for more information on Detailed Status.
- When one of the icons changes to yellow, it denotes a critical failure. Red denotes that a job did not print because of an error at the printer.

**Note:** Correct the printer error and print the entire job to change the red icon to green again.

**Note:** Printer 2 has an alias defined. This is helpful when reading the tree.



Status Menu Choices

File Menu	
Command	Description
Specify Server	Allows you to add another LPS server to the tree view. You must know the IP address of the server in order to do this. A better way would be to select all servers that you want to view from the server arbitration dialog box explained earlier. You may have to use this dialog box if your LPS server is on a different subnet. <div><b>Note:</b> Software support is limited to one print group per LPS server in the Status Dialog.</div>
Exit	This exits the application.

Options Menu	
Command	Description
Expanded	Expands or collapses the tree view on the left to include all server(s), printer(s) and Print Jobs (Printed, Pending and Critical Failures).
Expand Devices	Expands or collapses the tree view on the left to include all server(s) and printer(s).
Collapsed	Collapses the tree view on the left to include only the server(s).
Language	<p>This command opens a dialog that allows the user to change the language of the Status Application. The selection of the default language for Software applications is made during the original installation. However, there may be cases where the default language of Software applications needs to change with different users.</p> <p>For example, the application language is set to English during the install. This has worked well for the person doing the installation, User A, the next logged in user, User B, and the third logged in user, User C. However, Users D, E and F would like the language of the LPS Status Client application to be displayed in their native language, French. To change this setting for User D and the subsequent users, E and F in this case, perform the following steps:</p> <ol style="list-style-type: none"> <li>1. Press Options   Language from the menu bar.</li> <li>2. Select Français (French) from the drop-down list.</li> <li>3. Press and hold Ctrl, Shift and L simultaneously on the keyboard.</li> <li>4. Select <b>Set as Default Language for New Users</b></li> </ol> <p>This changes the default Status Client application language for Users D, E, and F to French, but it does not affect Users A, B, and C, whose Status Client language remains English. This setting may be changed as often as needed, but each previous logged in user's settings are retained unless subsequently changed in the Options   Language menu.</p>
Refresh	This command refreshes all of the counts for all of the servers and queues. There may be a slight delay if the server is busy when it receives this command. If a server goes offline (red) and then comes back on-line, clicking <b>Refresh</b> forces a re-connect immediately as opposed to waiting for the polling refresh. The server displays as green right away.

Filter Menu	
Command	Description
Pending	Clearing this option causes all pending jobs to be suppressed from the context view. This makes it easier to spot printed jobs and jobs with errors.
Critical Failures	Clearing this option causes all jobs with errors to be suppressed from the context view. This makes it easier to spot pending jobs and printed jobs.
Printed	Clearing this option causes all printed jobs to be suppressed from the context view. This makes it easier to spot pending jobs and jobs with errors.

Filter Menu	
Command	Description
Spooled	Clearing this option causes all spooled jobs to be suppressed from the context view. This makes it easier to spot printed jobs, jobs with errors, and pending jobs.

**Note:** The more jobs you filter, the faster, and less informative, is the tree. Filtering job types that do not interest you, such as Printed in many cases, can make a big difference in refresh speed, especially when several hundred printers are being monitored.

Help Menu	
Command	Description
Status Specific Help	Brings you to Status Client Application information in the LPS online help User's Guide.
Index	This brings you to the help index of the User's Guide.
About	This brings up an <b>About</b> window with the version number.

Toolbar	
Field	Description
Server	Drop-down list displays the server(s) to which you are connected.
Group	Drop-down list of the print groups assigned to the selected server.
Device	Drop-down list displays a specific printer, or all the printers connected to the chosen server(s) and selected print groups.
Printed Jobs	Number displayed is the number of jobs that have printed for the selected printer.
Pending Jobs	Number displayed is the number of pending jobs for the selected printer.
Critical Failures	Number displayed reflects the number of Critical Failures for the selected printer (s).
Spooled Jobs	Number displayed reflects the number of Spooled Jobs for the selected printer (s)

Context (right click) Menu	
Command	Description
Monitor Group	Only available when you right click on a selected server in the tree panel. Use this context menu to change the print group on the server you are viewing.
View File	Launches the File Editor, which opens the job's file, be it a PAS, CSV, XML or Command and Batch file. You can then edit the file and resubmit it to the LPS server. More information follows on the File Editor.

Context (right click) Menu	
Command	Description
Resend Job	Resends the selected job to the LPS server. Jobs marked as Pending cannot be resent.
Resend Selected	After performing a multiple selection, resends the selected jobs to the LPS server. Jobs marked as Pending cannot be resent.
Delete Job	Deletes the job from the LPS server along with its files.
Delete Selected	After performing a multiple selection, deletes the selected jobs from the LPS server along with their files.
Delete All	Deletes failed and printed jobs for whatever you have selected in the tree view.

**Related Information**

For more detailed information on Polling and screen refreshing, refer to *Understanding Screen Layout* in this guide.

## File Editor

The File Editor is launched by right clicking on a printed job on the Status dialog and selecting View File. It is used to make changes to a job's file and resend the job. This may be helpful when you are configuring your system, and you want to debug and fix your requests on-the-fly from the status program.

**File Editor**

File Name: 201109291620000000\_LPS001.pa Job Number: 5 Device Number:

Job Name: SampleJob002 Job Type: Pas File 2

\*FORMAT ZPLLabel  
\*JOBNAME,SampleJob002  
\*QUANTITY,1  
\*PRINTERNUMBER,2  
Barc0000,132456798  
Barc0001,loftware.com  
\*PRINTLABEL

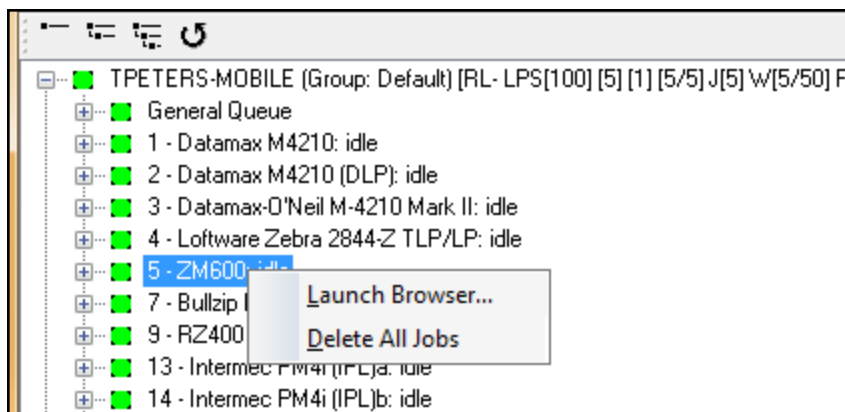
Find Resend Cancel

**Note:** To change the printer to which the job was assigned you MUST change the Printer Number edit box rather than the file, as the file itself is NOT re-processed.

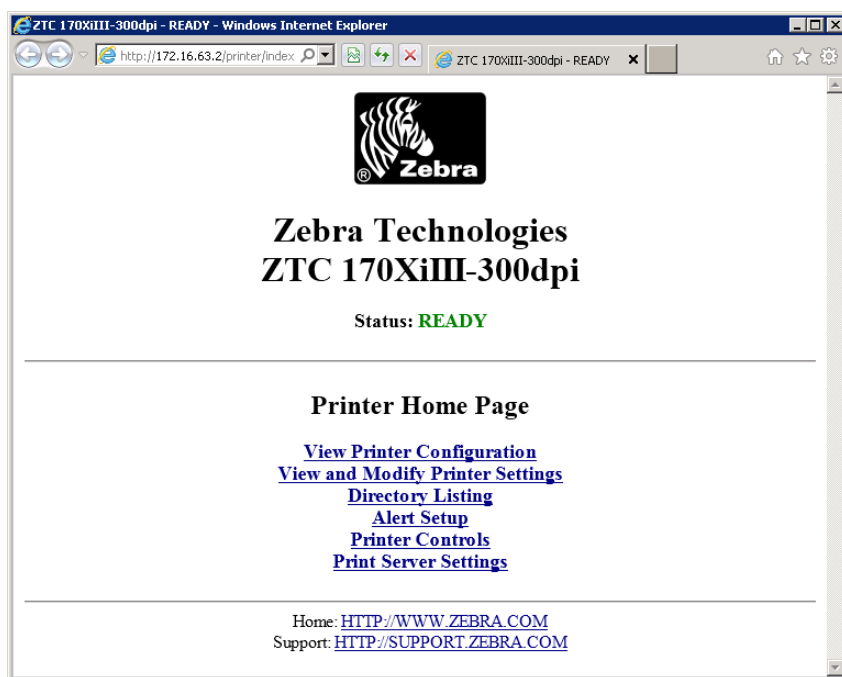


## Using the Launch Browser Feature

If a printer listed in the **Status** dialog window has been configured in Software with an IP address, you can open a Print Server Web Page for the printer in a browser window. To do this, right-click the printer and select **Launch Browser**.



The Printer Print Server home page is launched in a browser.



Click the links to view information or change configurations for the printer.

The following may help you to troubleshoot issues with the Software Print Server. Please go through this list before contacting Software's technical support department.

**Note:** Errors can be evaluated by reading the LPS entries in the Event Viewer. See the next section for information regarding these entries.

### LPS Service

- If you have trouble with the LPS running as a service, stop the service and try again in Interactive mode. This mode will provide error messages that you would not see running in Service mode.
- If the LPS service is running and you have made any changes to the way the hardware key is configured, you must stop and then restart the service in order for the changes to be recognized.
- If you receive an "Error: 2140" message explaining that the service could not be started, it could mean that the user or domain user account does not have Admin privileges on the local system.

### Logging

- After you are finished with the implementation/debugging phase of your project, make sure that you turn off **Event Logging** in the logging section of the **LPS Configuration utility**. Otherwise, your event log may be filled with messages that are unnecessary in production, and your hard drive could become filled with trace files.
- Make sure that purge settings are set in the **Housekeeping** tab of the LPS Configuration utility. The purge settings can always be optimized as you get used to your system and its performance.

### Scan Directories and File Drop

- Make sure that the LPS is not scanning the root directory of any drive. In addition, when the scan directory is set, it must already exist.
- Verify that there are no spaces before or after your label name on the "\*Format" line.
- If you are testing PAS, CSV, or XML files that you create in a text editor, be sure that you put the complete file name in quotation marks when you save from the text editor. For example, "testfile.pas", to avoid creating a file with the wrong extension that will not print when dropped. For example, testfile.pas.txt. Do not hide common file extensions in Windows. Select Tools | Folder Options | View Hide extensions for known file types from any window to make this issue easier to identify.

- If you are scanning a network drive or printing to network printers, the service must be configured to start up with a specific account that has access to all the network resources that need to be used. The account must have administrative privileges.
- If you are using PAS files, and you do not specifically want to print spaces at the leading or trailing end of any field, configure the LPS to trim Both in the control panel configuration utility.
- Do not set your scan directory to be the same as the \batch directory. The activity of creating and deleting the .tmp batch files causes LPS to needlessly check the directory.

## Emergency Mode

Emergency Mode allows you to run the Software Label Manager for a short period of time (10 printing days) after a key or password license failure during production. It is a temporary solution if you require immediate restoration of Software label printing operations during non-business hours for the eastern coast of the United States.

**Note:** Emergency Mode is only available if you purchase and license LPS.

When the Software Label Manager detects a key failure, a message is logged into the Windows Application Event log and the Software Label Manager shuts down. A hyperlink in the Windows Application Event log will direct the user to a page on the Software Web site with instructions on how to initiate and run the Software Label Manager in Emergency Mode. If you see these instructions in the Event Log, then Emergency Mode is available to you. At this point, you must follow the instructions outlined below to get up and running again if it is during non-business hours for Software. If you do not, the Software Label Manager will enter demo mode when you restart and data will be scrambled on your labels.

**WARNING:** Do not use Emergency Mode on the primary node of a cluster; LPS may not fail over in Emergency Mode. If the license fails on the primary node, it is best to fail over to the backup node. If the licenses on both nodes fail, then the Emergency Mode should be used to resume production.

While in Emergency Mode, Software applications will function as valid licensed applications. The Emergency Mode license is identical to the most recent valid license. The difference is that the Emergency Mode license will expire, and it will include specific logging and informational messages.

**Note:** Emergency Mode is a one-time mode. Make sure you contact Software Customer Service before the 10-day countdown expires for a reset of your original license.

## Emergency Mode Period

Emergency Mode eventually expires. The period counts down for each day that a print request is completed.

### For example

A 10-printing-day license for a company that does not have weekend production would last for 2 full business weeks.

With LPS running as a service, you will need to view the Event log to verify the number of printing days remaining. From the Software Label Manager, a message will indicate the number of printing days remaining on startup. Help | About will also show the number of printing days remaining.

## Emergency Mode Sequence

1. You experience either a key or a password failure. LPS has shut down and it has been verified from the Event log that a key or license failure has occurred.
2. You run the Gatekeeper.exe application, which can be found in the Software install folder - <Program Files Folder>\Software Labeling, by default. You apply an Emergency Mode license by selecting **Generate Emergency License**.

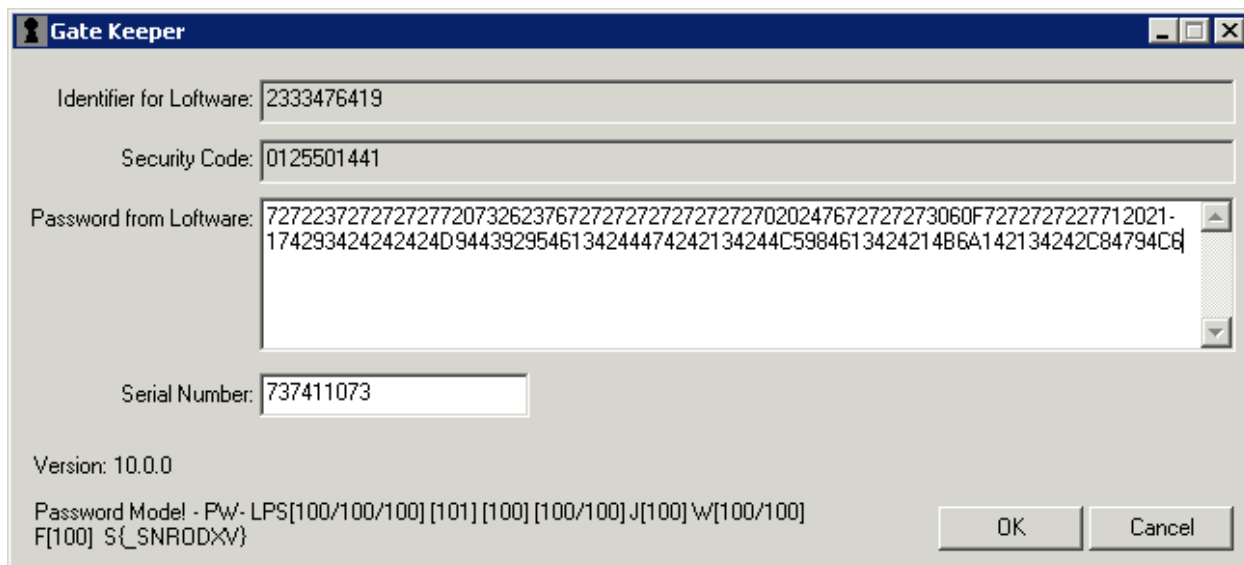


Figure 7.13: Gatekeeper generating emergency license

3. Run LPS in Emergency Mode until Software opens for business. If the Emergency Mode password fails, you will need to fail over to your backup server if you have one set up.
4. Contact Software Customer Service to request a permanent replacement password or to initiate the key replacement process.
5. Software Customer Service will issue a new password or replacement key.
6. You run Gatekeeper and apply the new password, or the you install the new key.

## When Emergency Mode Expires

If the LPS is running, it will shut down when the first job is submitted after the LPS detects the expiration. Until a valid license is applied, Software applications will run in demo mode. A new Emergency Mode license cannot be issued.

## **Getting out of Emergency Mode**

Loftware Customer Service assistance is required for a system in Emergency Mode to resume running on a valid hardware key or password license. If hardware key failure is the problem, you must contact Loftware Customer Service on the next business day to start the key replacement process. If password failure is the problem, you must contact Loftware Customer Service to request a permanent replacement password.

## **Resuming Printing in Full Mode**

When you apply a new password or key to your LPS system in Emergency Mode or after Emergency Mode has expired, LPS will resume normal operations.

## Software Utilities

---

Software provides the following utilities to help with configuration, troubleshooting, and testing.

- ["Software Version Utility" on page 118](#)
- ["LPSSend Interface" on page 119](#)
- ["WDPing Diagnostic Utility" on page 126](#)
- ["Software Reporter Utility" on page 131](#)
- ["Backup and Restore Utility" on page 138](#)

### Software Version Utility

The Software Version Utility (LWVersionUtility) is a command line tool used to promote Software labels from the current instance to another instance. For example, from a Development environment to a Production environment.

LWVersionUtility uses the Instance configuration information set in LWVersion.ini. For more information, see ["Enable Label Versioning" on page 44](#).

#### Options

##### Original

Specifies the path to the file to be promoted. UNC paths may be used.

##### Syntax

```
-Original="<Label Location>"
```

##### Instance

Specifies the instance to which to promote the label.

##### Syntax

```
-Instance="<InstanceName>"
```

##### RevisionComments

Specifies comments to include with this version of the label.

##### Syntax

```
-RevisionComments"<Comment text>"
```

## Promote a Label From the Command Line

See the *Loftware Label Manager User Guide* for information on promoting labels from within Design.

1. Open a Command Prompt and navigate to the Loftware Labeling folder.

```
C:\Program Files (x86)\Loftware Labeling>
```

2. Use the `lwversionutility` command to promote a label. The following example will promote a label named `label1.lwl` from a local development instance to production with a comment.

```
C:\Program Files (x86)\Loftware Labeling>lwversionutility -  
original="C:\Program Files (x86)\Loftware Labeling\Labels\label1.lwl" -  
instance="Production" -revisioncomments="Ready for production printing"
```

The following appears upon successful promotion

```
INFO: Label promotion process started.  
INFO: Reading source file..  
INFO: Started reading destination file..  
INFO: Updating the destination file label1.lwl with Revision 1  
  
SUCCESS: Label promoted successfully.
```

## LPSSend Interface

The LPSSend interface can be used to demonstrate the TCP/IP Socket Interface of the Loftware Print Server. This interface opens a socket connection to the LPS, sends a print job, receives status of the job from the LPS and then disconnects. The source code for this utility is included and is available for your use.

**Note:** This section is for advanced users and assumes you are familiar with terminology and usage of command prompts and programming.

LPSSend.exe and its source files are located in the <Program Files Folder>\Loftware Labeling\Sample Programs\LPS\_Send directory. Refer to the *LPSSend\_AD1.doc* file for detailed programming information. This document also contains important information for passing data back to the calling program; like EPC RFID data.

## LPSSend Files

- `lpssend.cpp` is the actual code and implementation of `lpssend`. If you want to view the code behind `lpssend`, open it in Microsoft Visual Studio or another editor.
- The three `.h` files are the header files for the `lpssend` project. They are required for building of `lpssend` to be successful.
- The `.dsp` file is the `lpssend` project file for use with Microsoft Visual Studio.

- The .dsw file is the lpssend workspace for use with Microsoft Visual Studio which contains the LPS .dsp protect as well as the external dependencies (.h files)
- The .exe file is the lpssend executable.

The following steps are designed to give you some insight into how LPSSend works. Much of the code in LPSSend can be re-used by your own application. Please take the time to understand how LPSSend works. After the steps are fully understood, you are able to “cut out” the parts that you need and “paste” them into your own code.

**Important Note:** Remember that the socket interface to the LPS is synchronous in nature. Your program **MUST ALWAYS WAIT** for a confirmation response back from the LPS before proceeding. This applies for Log In and Log Out as well as print requests

## Step 1 - Obtain LPS Address

The first step of LPSSend is to read in the arguments or parameters from the command prompt. Once these parameters have been read in, LPSSend uses the IP address and port supplied to open a socket connection to the LPS. Your program may hard code these values, or get them from a different place.

## Step 2. Log into LPS

Once LPSSend determines that it has opened a socket connection to the LPS, a login request is sent. LPSSend sends the Computer name, User name, and version number to the LPS requesting a login. The actual protocol used to request this login is also sent. The code is: REQ\_LPSLOGIN.

## Step 3. Wait for Login Confirmation

After the login request is sent, LPSSend waits for a response from the LPS. The LPS sends back another protocol command to the LPSSend client: RSP\_LPSLOGIN. LPSSend is successfully logged into the LPS.

## Step 4. Prepare Print Request

Now that LPSSend is logged in, it prepares the Print job request. LPSSend creates a buffer large enough to store the pas, csv, or xml file specified in the command prompt parameters. The file is read into the buffer, and packaged with the printer number or printer alias, job type (PAS, CSV or XML), job name (if specified), as well as the protocol command: REQ\_SNDJOB.

## Step 5. Send Print Request, Wait for Response

When a print job has been sent to the LPS, LPSSend waits for the LPS to finish processing the job and send back a response. The LPS sends the protocol command RSP\_JOBUPDATES, along with some extra data about the job. The following are possible responses to a sent job:

- Printed
- Critical Failures



- Pending
- Spooled

LPSSend displays this information for the user to acknowledge the status of the job sent. Your program should handle the responses in an appropriate manner.

**Note:** Repeat Steps 4 and 5 as many times as needed without logging out.

## Step 6. Logout Request

Once LPSSend's job has been sent and processed, LPSSend logs out from the LPS, with the protocol command: REQ\_LOGOUT.

## Step 7. Wait for Logout Confirmation

LPSSend waits for a response to the log out request from the LPS. The LPS sends back the protocol command RSP\_LOGOUT.

## Step 8. End

Once LPSSend has received its response to the logout request; LPSSend closes the socket connection and frees the buffer that was allocated for the print job request.

## LPSSend Usage

LPSSend.exe is a sample program that contains all the documentation needed for TCP/IP connections. You can use LPSSend to learn about socket connections. Running LPSSend is a great way to test these connections, and Loftware strongly recommends that you create a simple file in order to do this.

Run LPSSend from a command window using the following information as arguments:

```
LPSSend Address -pPort [2723] -tJobType [0]
-nPrinterNumber |-aPrinterAlias -jJobName -fJobFile
(The numbers in brackets [] are default values.)
```

Options	
Address	Specify the LPS IP address
-p	Specify the port (default: 2723)
-t:	JobType
-t0	PAS File (default)
-t1	CSV File
-t2	XML File
-n	Loftware Printer Number

Options	
-a	Software Printer Alias (Either Number or Alias is sent)
-j	JobName Name of Job
-f	Filename of job to send (with path)
-r	(By itself) Displays a chart of the return code values
-h	(By itself) Displays this help (also displayed by -?)
-l	Login type [0] default - log in only, no request made as yet
-l1	Login, send job request, wait for status
-l2	Login, send job request, no wait for status
-s	Status requests

## LPSSend Usage Examples

### Example 1

```
lpssend 172.16.0.64 -n1 -flabel1.pas
```

This example sends the label1.pas file, located in the same directory as lpssend.exe, to the IP address 172.16.0.64 with default port of 2723 and default jobtype of 0 (.pas file) printed on Printer 1.

### Example 2

```
lpssend 172.16.0.64 -t1 -n2 -f "C:\Program Files (x86)\Software  
Labeling\Batch\label1.csv"
```

The above example sends the label1.csv file, located in the Software Labeling Batch directory, to the IP address 172.16.0.64 with default port of 2723 and a jobtype of 1 (.csv file) printed on Printer 2.

## LPSSend Label Example

For this example, a label called "lwlabel.lwl" is created in Software Label Manager Design for a Datamax Titan 6200 printer, configured as Device #8.

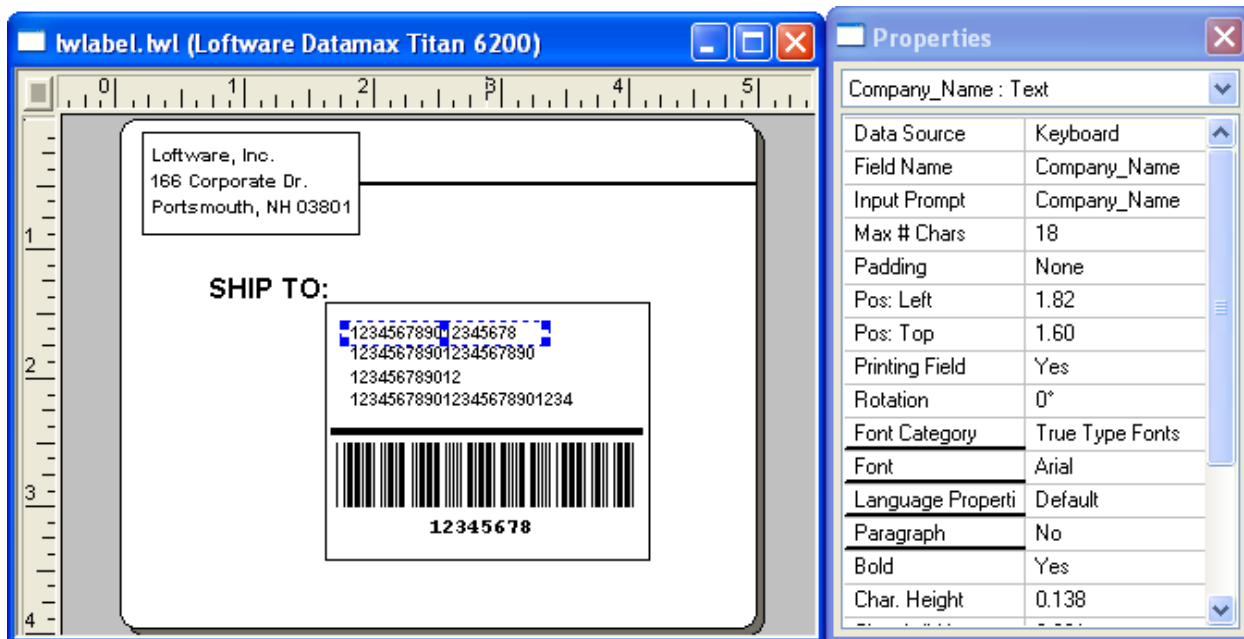


Figure 7.14: lwlabel.lwl

A .pas file called lwlabel.pas is created for this label and saved to the TCPIPtLPS\_Request directory. (If your program builds a .pas file in memory, do not forget the <CR> <LF> at the end of each line.)

```
*FORMAT,lwlabel.lwl
*JOBNAME,SampleJob029
*QUANTITY,1
*PRINTERNUMBER,8
COMPANY_NAME,Aztec Corporation
ADDRESS_1,1221 Inca Dr
ADDRESS_2,P.O.Box 17
CITY_STATE_ZIP,Yuma,AZ.63325
Bar Code,552-300
*PRINTLABEL
```

**Note:** If the file is located in any directory other than TCPIPtLPS\_Request, you must provide the complete path to the directory. For example, C:\Program Files (x86)\Software Labeling\Batch\lwlabel.pas.

The command window displays the result of the LPSSend example.

```

C:\Program Files\Software Labeling\Sample Programs\TCPIPtLPS_Request>lpssend 172.16.0.68 -n8 -flwlabel.pas
*LPSSend: Connected to "172.16.0.68" on port 2723.
*LPSSend: Sent Log In Request.
*LPSSend: Received Log In Response from "RICA-2000".
*LPSSend: Sent Print Job Request.
*LPSSend: Received Status: "Printed" for Job "28" on Printer "8".
*LPSSend: Sent Log Out Request.
*LPSSend: Logged Out.

C:\Program Files\Software Labeling\Sample Programs\TCPIPtLPS_Request>lpssend 172.16.0.68 -n8 -flwlabel.pas
*LPSSend: Connected to "172.16.0.68" on port 2723.
*LPSSend: Sent Log In Request.
*LPSSend: Received Log In Response from "RICA-2000".
*LPSSend: Sent Print Job Request.
*LPSSend: Received Status: "Printed" for Job "29" on Printer "8".
*LPSSend: Sent Log Out Request.
*LPSSend: Logged Out.

```

Figure 7.15: Command Window displaying the LPSSend information

The label requested above prints out as displayed below:

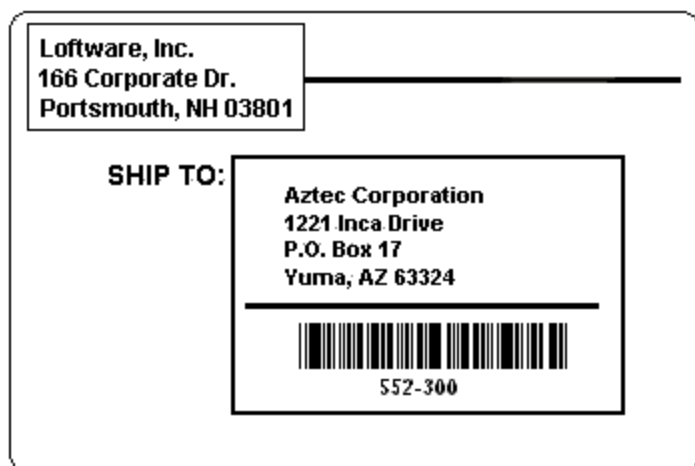


Figure 7.16: Completed Label using LPSSend

## LPSSend Return Codes

LPSSend returns a code at the end of execution. The following chart explains the meaning of each of these return codes.

### Return Codes for Main

GENERAL MAIN EXITS	
101	Display Return Codes
102	Display Args
103	Invalid Args

<b>GENERAL MAIN EXITS</b>	
104	Missing Args
105	File Name or Path not found (of the pas, csv or xml file)
<b>SOCKET ERRORS</b>	
201	Error on Send Login Request
202	Error on Receive Login Response
203	Error on Send Print Job Request
204	Error on Receive Print Job Response
205	Error on Send Logout Request
206:	Error on Receive Logout Response
207:	Error on Send Login Send Wait, Login Send Wait XML or Login Send No Wait Request
208:	Error on Receive Login Send Wait, Login Send Wait XML or Login Send No Wait Response
<b>RECEIVED OUT OF FRAME DATA (MAGIC NUMBER)</b>	
301	Receive Login Response
302	Receive Print Job Response
303:	Receive Logout Response
304:	Receive Response to Login Send Wait, Login Send Wait XML or Login Send No Wait
<b>FAILURES</b>	
401	Failed to Open Server
402	Login Failed
403	Print Job Failed
404	Logout Failed
405	Login Send Wait Failed
406	Login Send No Wait Failed
407	Login Send Wait XML Failed
408	Print Job XML Failed
<b>END OF MAIN (STATUS OF JOB)</b>	
501	Critical Failure

END OF MAIN (STATUS OF JOB)	
502	Printed
503	Printed with Errors
504	Printer Error
End of Main Returns 0 (should never happen)	

## About LPSSend Client

The LPSSend interface can be used to demonstrate the TCP/IP Socket Interface of the Software Print Server. This interface opens a socket connection to the LPS, sends a print job, receives status of the job from the LPS and then disconnects. The source code for this utility is included and is available for your use.

## WDPing Diagnostic Utility

The WDPing utility allows you to check the status of your printing systems from across the network without disturbing your labeling system in any way.

This utility is installed in the <Program Files folder>\Software Labeling subdirectory by default. When invoked from a command prompt, it displays information on screen based on the parameters supplied. Issuing the WDPing command with no parameters displays the usage syntax parameters.

## To use WDPing

1. Open a Command Prompt.
2. Type "cd <Program Files folder>\software labeling". Press **Enter**.
3. Type "wdping -i [LPS IP Address]"

**Note:** You can type -? to display all the useable parameters in the Software Diagnostic Utilities. Command line options do not need to be in any specific order.

### Example

"wdping 172.16.0.9 -u" is the same as "wdping -u 172.16.0.9"

**Note:** Many of the following settings are used by Software technicians for troubleshooting purposes. End users of the Software Print Server commonly use "Broadcast" and "Status."

## WDPing parameters

**Note:** If, for security reasons, you do not want your server to respond to a ping from one of these utilities or a broadcast from an LPS client program, call Software Technical Support for assistance with this advanced option.

## Broadcast (-b)

Broadcasts a “find LPS servers” message across the network and returns the ComputerName and IP Address of any Loftware servers that are currently running.

### Command Syntax

```
WDPing -b
```

### Output

```
Server SHIPPING located at 172.100.0.1  
Server RECEIVING located at 172.100.0.2
```

## Status (-s)

Sends a status request to a specific LPS server identified by an IP Address. This IP address is obtained from performing a “-b” broadcast as mentioned previously. If the server is running, the following is returned:

- Round trip time for the LPS response
- The computer name where the LPS resides
- The number of currently connected On-Demand Print Clients
- The High Water Mark (HWM) indicating the maximum number of clients connected at any one time since the LPS service was started.

### Command Syntax

```
WDPing -s 172.100.0.1
```

### Output

```
Response received. Roundtrip time 19 Milliseconds.  
SHIPPING has 5 connected clients. HWM is 11
```

## Client Licenses (-i)

Retrieves the Client Licenses in use by the Loftware Print Server. You must include the IP address of the server.

### Command Syntax

```
WDPing -i 172.100.0.1
```

### Output

```
LPS Client License report  
Client License Type: Standard LPS Clients  
Max Clients: 2  
In Use : 0
```

```

Client Use Count
=====
=====
Client License Type: Java Connector Clients
Max Clients: 2
In Use : 0
Client Use Count
=====
=====
Client License Type: Oracle Connector Clients
Max Clients: 2
In Use : 0
Client Use Count
=====
=====
Client License Type: Software Connector for SAP Clients
Max Clients: 2
In Use : 0
Client                               Use Count
=====
=====
Client License Type: Web Service Licenses
Max Clients: 2
In Use : 0
Client Use Count
=====
=====
Client License Type: LWA Session User Licenses
Max Clients: 2
In Use : 0
Client Use Count
=====
=====

```

## Client Information (-c)

Sends a request for Client information to a specific LPS service identified by an IP address.

## Command Syntax

```
WDPing -c 172.100.0.1
```

## Output

ID	ComputerName	User	Address	Port	Last Activity
00000001	SHIP_PC_1	SHIPPER1	172.100.0.100	39172	19990924 11:00:18

The ID is assigned to the Client by the LPS when the client connection is initiated. The Computer Name is the name of the computer where the client is running. User is the user currently logged onto the Client computer. Address is the IP Address of the Client, Port is the TCP port connection for the server and the client, and Last Activity is generally the time that the client last printed a label.



## CPU Usage (-u)

Sends a request for CPU Usage information to a specific computer running the LPS service. If the service is running, the following information is returned:

### Command Syntax

```
WDPing -u 172.100.0.1
```

### Output

```

Administrator: Command Prompt
Service start: 10/13/2011 12:18:47
Running Time : 1 Days, 05 Hours, 06 Minutes, 24 Seconds
User Time   : 000:00:00
Kernel Time : 000:00:01
CPU Usage   : 000:00:02
Avg CPU Usage: 0.00%

Thread
=====
BPTThread
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
DirMgrThread
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
DIR1
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
SockSvcThread
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
WorkerThread 0
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
WorkerThread 1
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
WorkerThread 2
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
WorkerThread 3
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
WorkerThread 4
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
DGRAMThread
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
PurgeThread
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
StatusManagerThread
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
StatusInterfaceThread
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00
CWDMemLogThread
      (User) 000:00:00
      (Kernel) 000:00:00
      (Total) 000:00:00      0.00      0.00

```

**Note:** Most of the returned settings are for technician use only.

## Default Response time (-w)

Used in conjunction with other WDPing command line parameters, this sets the time in seconds that the WDPing utility waits for a response from the LPS service. The default value is 2 seconds. Under normal circumstances, a response is received well within the default time. The only time you may need to increase the default value is if network traffic is heavy.

### Command Syntax

```
WDPing -s 172.100.0.1 -w10
```

## Thread Status (-t)

**Note:** This setting is for technician use only.

This sends a request for thread status to a specific LPS server identified by the IP Address supplied on the command line. If server is running, the following information is returned:

### Command Syntax

```
WDPing -t 172.100.0.1
```

### Output

```
BPThread Running
DirMgrThread Running
DIR1 Running ( 0 Files Processed)
SockSvrThread Running
WorkerThread 0 Active IDLE [1] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] {1}
WorkerThread 1 Active IDLE [1] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] {0}
WorkerThread 2 Active IDLE [1] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] {3}
WorkerThread 3 Active IDLE [1] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] {2}
WorkerThread 4 Active IDLE [1] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] [0] {3}
DGramThread Running
LogThread Inactive
PurgeThread Running (HK-OFF No Purges to Report)
(Purge Count: 0)
StatusManagerThread Running
StatusInterfaceThread Running
CWDMemLogThread Running
```

## Output Capture (-o)

Used in conjunction with other WDPing command line parameters, this specifies a file name for capturing the data that is returned by the various WDPing requests. If no file name is specified, data is captured in "<Program Files folder>\Software Labeling\wdping.txt".

### Command Syntax

```
WDPing -s 172.100.0.1 -oLPS.LOG
```

## Memory Use (-m)

```
Memory reset feature is enabled.
Interval : 24.0 Hours
Threshold : 0.0 KB
Factor : 2.00
Reset Count : 0
PID  ProcessName      Mem Usage  VM Size  Total
==== =====
432  WatchDogNT.Exe     14.4 MB   4.5 MB   18.9 MB
2636 LLMWBP32.EXE      23.6 MB   16.4 MB   40.0 MB
```

## Loftware Reporter Utility

The Loftware Reporter Utility (LWReporter) is a command line tool used to gather and report information about your Loftware Print Server or Loftware Label Manager system. LWReporter creates separate XML files that describe each file or component of your system.

### Reports Folder

The XML files generated by LWReporter are output to the <Program Files Folder>\Loftware Labeling\LPSReports folder by default. The output location can be changed using the -output option.

### XSD Files

The LWReporter includes XML Schema Documents (XSD) that define the elements and information contained in the various XML files generated. You can use the XSD files to create your own XSLT files. XSLTs can be used to convert the LWReporter XML output into customized reports in almost any format including text, PDF, CSV, or HTML. The XSD files are located within the <Program Files Folder>\Loftware Labeling\LPSReports folder.

For more information on XSLT, see the [XSLT Standard](http://www.w3c.org) at w3c.org and learn how to use [XSLT Transformations](http://www.w3schools.com) at w3schools.com.

## To use LWReporter

LWReporter is a command line utility.

1. Open a Command Prompt window.
2. Run the following command to navigate to the **Loftware Labeling** folder:

```
cd\<Program Files folder>\Loftware Labeling
```

### Example

```
cd\Program Files (x86)\Loftware Labeling
```

3. Enter lwreporter followed by inputs or options as described in the following sections.

**Note:** You can type -? to display all the useable parameters of the Loftware Reporter Utility. Command line options do not need to be in any specific order.

## Reporter Inputs

LWReporter can create XML reports on the following Software Print Server or Software Label Manager components. The Reporter Utility may fail processing certain files. If the file cannot be opened in Design, then it cannot be processed using LWReporter. It reports the names of files that cannot be processed. Non-Software file types are ignored by LWReporter.

**Note:** The default locations specified in this section may be different if you have installed to a different location or changed the location via Options | File Locations in Software Label Manager Design.

## Configurations

Locates and reports the settings in the Software Print Server and Label Manager INI files and the Windows registry. An XML file is created that corresponds with each INI file, and a single file is created with all the registry entries inserted by the Software Print Server.

### Syntax

```
lwreporter -configurations
```

## Field Lists

Locates and reports the field list (LST) files available in the <Program Files Folder>\Software Labeling folder.

**Note:** This location cannot be changed using Options | File Locations.

### Syntax

```
lwreporter -fieldlists
```

## Labels

Locates and reports the label (LWL) files you have created at the root of the <Program Files Folder>\Software Labeling\LABELS folder.

### Syntax

```
lwreporter -labels
```

## Layouts

Locates and reports the label layout (LWY) files located in the <Program Files Folder>\Software Labeling\Layouts folder.

### Syntax

```
lwreporter -layouts
```

## Queries

Locates and reports the query (LWQ) files you have saved from the Range Print Query Assistant to the <Program Files Folder>\Loftware Labeling\query folder.

### Syntax

```
lwreporter -queries
```

## Serials

Locates and reports the serial (LWS) files located in the <Program Files Folder>\Loftware Labeling\Serial folder.

### Syntax

```
lwreporter -serials
```

## Templates

Locates and reports the template (LWT) files located in the <Program Files Folder>\Loftware Labeling\Templates\General folder.

### Syntax

```
lwreporter -templates
```

## Printers

Locates and reports the configured printers including printer options and connection information.

### Syntax

```
lwreporter -printers
```

## Import Printers

Validate the device information in the **printr32.xml** file to be imported to LPS and LLM. The default location for this file is **Loftware Labeling\PrinterImports**. For more information, see *Importing Multiple Devices* in the *Loftware Label Manager User Guide*.

### Syntax

```
lwreporter -importprinters
```

## Reporter Options

The following options can be used with the LWReporter command:

## Commit

Import the device information in the **printr32.xml** file to LPS and LLM. The default location for this file is **Loftware Labeling\PrinterImports**. For more information, see *Importing Multiple Devices* in the *Loftware Label Manager User Guide*.

### Syntax

```
lwreporter -importprinters -commit
```

## Filter

Sets a wildcard-supported string to match against input files.

### Syntax

```
-filter=<filter text>
```

### Example

```
lwreporter -labels -filter=label*.lwl
```

## Output

```
Processing... C:\Program Files (x86)\Loftware Labeling\labels\Label1.lwl
Processing... C:\Program Files (x86)\Loftware Labeling\labels\Label2.lwl
Processing... C:\Program Files (x86)\Loftware Labeling\labels\Label3.lwl
Processing... C:\Program Files (x86)\Loftware Labeling\labels\Label5.lwl
```

## Input

Sets a specific location to report on. By default LWReporter uses the locations defined in Options | File Locations.

### Syntax

```
-input=<location>
```

### Example

```
lwreporter -labels -input=labels\region1
```

## Result

```
Processing... C:\Program Files (x86)\Loftware
Labeling\labels\region1\Label1.lwl
Processing... C:\Program Files (x86)\Loftware
Labeling\labels\region1\Label2.lwl
Processing... C:\Program Files (x86)\Loftware
Labeling\labels\region1\Label3.lwl
Processing... C:\Program Files (x86)\Loftware
Labeling\labels\region1\Label5.lwl
```

## Output

Sets the location where the XML files are created. This can be a full or relative path.

**Note:** When the Configurations or Printers inputs are used, only the Output option is available.

## Syntax

```
-output=<location>
```

## Example

```
lwreporter -labels -output=LPSReports\labels
```

## Result

```
An XML file is created in the <Program Files Folder>\Loftware
Labeling\LPSReports\labels folder for each of the LWL files in the <Program
Files Folder>\Loftware Labeling\LABELS folder.
```

## Recursive

Repeats the report on all sub folders within the default or specified Input.

## Syntax

```
-recursive -output=<location>
```

**Note:** You must use the -output option with the recursive option. You can specify the default location, but it must be specified, or LWReporter will be unable to process the files.

## Example

```
lwreporter -labels -recursive -output=LPSReports\All Labels
```

## Result

```
An XML file is created in the <Program Files Folder>\Loftware
Labeling\LPSReports\All Labels folder for each of the LWL files in the
<Program Files Folder>\Loftware Labeling\LABELS folder and sub folders.
```

## Verify

Confirms the existence and validity of the input file or files without generating corresponding XML files.

### Syntax

```
-verify
```

### Example

```
lwreporter -labels -verify
```

Using this input and option you can identify corrupt labels that cannot be opened.

### Result

```
Processing... C:\Program Files (x86)\Software Labeling\labels\Label1.lwl
Processing... C:\Program Files (x86)\Software Labeling\labels\Label2.lwl
Processing... C:\Program Files (x86)\Software Labeling\labels\Label3.lwl
Processing... C:\Program Files (x86)\Software Labeling\labels\Label5.lwl

Reporting Errors:
Unable to process file: C:\Program Files (x86)\Software
Labeling\labels\Label6.lwl

Total Number of 'Labels' processed: 6
    Failed: 1
    Successful: 5
```

## Examples

### Reporting on the entire Template Folder

```
lwreporter -templates -input=C:\Program Files (x86)\Software
Labeling\Template -recursive -output=LPSReports\template
```

This command scans the entire <Program Files Folder>\Software Labeling\Template folder, creates an XML file for each template found, and outputs the files to a sub folder under LPSReports.

**Note:** The input option is used to move to the root of the Template folder which is different than the default location. The recursive option used to report on all sub folders must be used with the output option.



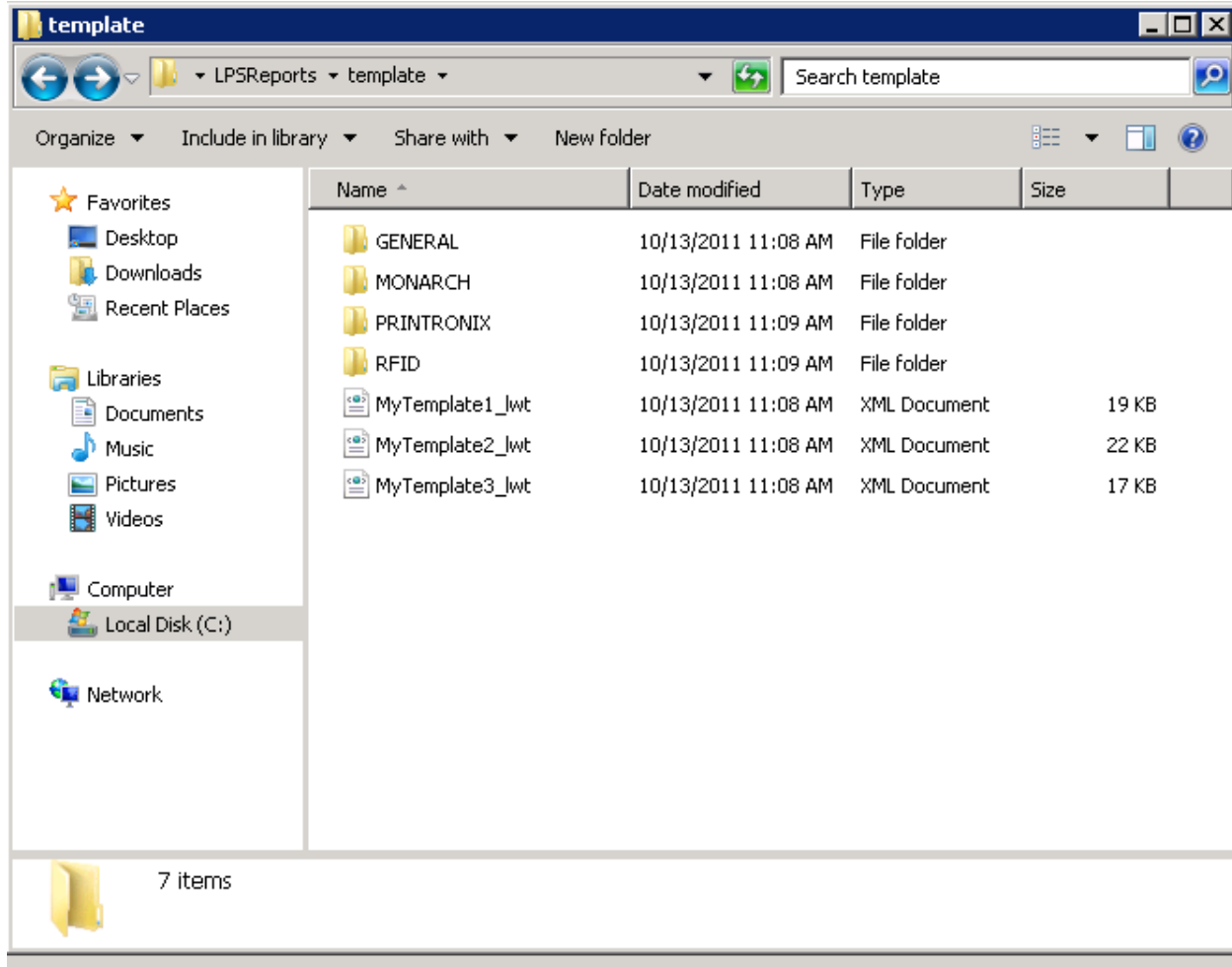


Figure 7.17: View of LPSReports\template folder after running lwreporter with Templates input

## Reporting on Labels Containing Specific Text in a Specified Folder

```
lwreporter -labels -filter="*east" -input="C:\Regional Labels" -recursive -
output="LPSReports\East Region Labels"
```

**Result**

```

Starting generation of Software Reports
This may take a few minutes...

Processing... C:\Regional Labels\LabelEast1.lwl
Processing... C:\Regional Labels\LabelEast2.lwl
Processing... C:\Regional Labels\LabelEast3.lwl

Reporting Errors:

Total Number of 'Labels' processed: 3
    Failed: 0
    Successful: 3

```

This command scans the C:\Regional Labels folder and sub folders for any labels that contain the word "east" within the label name. It then creates an XML report for each of the labels found and outputs the files to the C:\<Program Files Folder>\Software Labeling\LPSReports\East Region Labels folder.

## Backup and Restore Utility

LPS\_SYNC.EXE is designed to synchronize your LPS production configuration files and those of a backup, making the most recent backup files available when the need arises. It can be run from a command line to back up your LPS configuration files and then restore these files as required. You can back up files on Server A and restore them on Server B, keeping Server B in "sync" with Server A. How often you back up your system is proportional to how often you change its configuration settings. It can be run as a scheduled task, as described later.

Be aware that this utility and the procedures discussed in this section are a low tech / low cost means of keeping two server configurations synchronized.

### Important Notes about the Backup and Restore utility

**Important!** Software strongly recommends that you back up all production LPS systems with a third party back up solution.

The Backup and Restore utility (LPS\_SYNC.EXE) is only available with an LPS license.

- Restored settings will not take effect until the LPS is restarted. It is best to leave the LPS shutdown on the backup server until it is needed.
- The Backup and Restore utility will NOT back up your label data. Files that are backed up are listed below. You may want to consider having the backup and restore servers accessing label, serial, and image files on a central network drive.
- The Backup and Restore utility does not support UNC paths. Use a mapped drive if you are backing up or restoring from a network or a shared resource.

- If you are restoring from one server that is remapped to the registry to a second server that is not remapped, the second server will automatically be remapped. This will cause loss of any settings set in the ini files.
- If you are restoring to a server that uses mapped drive designations or paths different from the backup server, you will have to manually correct this.
- We strongly recommend that you use Backup and Restore on servers with the same operating systems.
- Do not use Backup and Restore with Clustering.
- Do not use Backup and Restore with Terminal Servers running LPS.
- We strongly recommend that the LPS backup server is installed in the same language as the production server.

## Mixed Versions

Backup and Restore will issue a warning when an attempt is made to back up on one version and then restore on a different version.

- Backing up and restoring from one major version to another (for example, 7.1 and 8.3) is NOT supported.
- Backing up and restoring within the same major version (for example, 8.1 and 8.3) is allowed after the warning is displayed and you choose to restore to a different version. However, while this is possible, it is discouraged. We strongly recommend backing up and restoring of matching versions only.

## Configuration

- The default location for the resulting archive file is Software Labeling\backups. You may override the default by specifying a different folder. Please see the table in the Using Backup section.
- The default location for restored files when the /r switch is used is the Software Installation directory (\Software Labeling). The default location for restored files when the /u switch is used is Software Labeling\backups\safe restore. You may override the default by specifying a different folder. Please see the table in the Using Restore section.
- You may specify a filename to back up to or restore from. Otherwise, Software naming conventions will apply. Please see the tables in the next sections.
- The application LPS\_Sync.exe runs from the command line using parameters described in the tables below.
- The application saves backups using the LPZ extension.

The following files are backed up:

- All LPS INI files, including but not limited to: DTagCfg.ini, llmcluster.ini, llmgrid.ini, llmthred.ini, llmwbp32.ini (for any defined Printer Groups), llmwclnt.ini, llmwcdn32.ini, llmwod32.ini, llmwrp32.ini, lps\_sync.ini, lwtabinfo.ini, lwversion.ini, printr32.ini

- User database entries: user32.lsv
- Printer Group files: instance32.id2, instance32.ld2
- Device configuration files: printr32.cfg, printr32.ini, printr32.lfg
- Software, Inc.reg and WatchDogNT.reg (note that a restore switch parameter can be used with LPS\_Sync so that registry files are not automatically restored. Instead, a .bat file will be created which, when run, will ask you if you wish to restore the registry files. See the /u and the /g parameters in the Using Restore section of this topic for more information.)

## Using Backup

Before using the backup feature of LPS\_Sync, make sure that the LLM Device Configuration grid is not open. To use the backup feature of this utility, from the command line, run LPS\_Sync.exe followed by a command line parameter, described in the following table:

Backup Command Line Parameter	Functionality
/b	Backs up files listed in the previous section to an .LPZ file. The default location for the resulting archive file is Software Labeling\backups. You may override the default by specifying a different folder. See examples in the next table.
/f"path\filename"	Optional. Allows the user to configure a path and file name for the resulting archive.
/s	Optional. Silent mode – No on screen messages are logged.
/l	Optional. This creates a log; default is Software Labeling\backups\logs\LPS_Sync.log
/y	Optional. Override questions with 'yes' as the default response
/h (or /?)	Displays options.

Following are examples of using command line backup parameters with LPS\_Sync.exe

Backup Scenario	Command	Result
Back up to default path using default naming convention.	LPS_Sync /b	File is created as Lps_Sync_200608011055.lpz in Software Labeling\backups, where 200608011055 is the date and time.
Back up to default path using unique naming convention.	LPS_Sync /b /f"ABCInc062606.lpz"	File is created as ABCInc062606.lpz in Software Labeling\backups\.
Back up to unique location using unique naming convention.	LPS_Sync /b /f"c:\temp\ABCInc062606.lpz"	File is created as ABCInc062606.lpz in c:\temp\ folder.

Backup Scenario	Command	Result
Back up to unique location using default naming convention.	LPS_Sync /b /f"c:\temp\"	File is created as LPS_Sync_200608011055.lpz in c:\temp\ folder.
Back up to default path using default naming convention – silent mode	LPS_Sync /b /s	The result of this command is similar to the first scenario previously mentioned. The difference is that no on screen messages will be displayed.

## Using Restore

Before restoring from a backup, shut down all LPS applications. To use the restore feature of LPS\_Sync, from the command line, run LPS\_Sync.exe followed by a command line parameter, described in the following table:

Restore Command Line Parameter	Functionality
/r	Restores a previously backed up lpz file. The default location when /r is used is the Loftware install folder. You may override the default by specifying a different folder where the files are sent to. See the example in the next table.
/f"path\filename"	Allows the user to configure a path and/or file name to restore from.
/l	Optional. This creates a log; default is Loftware Labeling\backups\logs\LPS_Sync.log
/u"path\"	This will unzip the files to the designated directory without restoring system files. Instead, registry files are placed in the unzip directory. A file called "registry_restore.bat" will be created in the directory where the files are extracted to. When run, the bat file will ask you if you wish to restore your registry files. If you respond with Yes, LPS_SYNC.EXE is run with a /g against the same archive that the safe restore was run on.
/g	This will safely restore only the registry from an archive. This uses the most recently created archive in the default archive directory, unless you specify a different archive using the /f option.
/s	Silent mode – No on screen messages are logged.
/y	Override questions with 'yes' as the default response
/h (can also use /?)	Displays options

Following are examples of using restore command line parameters with LPS\_Sync.exe.

Restore Scenario	Command	Result
Restore from default path	LPS_Sync /r	If there is more than one LPZ file in the default path, the most recent backup is restored.
Restore a uniquely named backup file (.lpz) from default path	LPS_Sync /r /f"ABCInc062606.lpz"	The contents of file ABCInc062606.lpz from default path are restored
Restore a default backup file from the default location to a unique location	LPS_Sync /r"c:\temp\"	The contents of the most recent lpz in the default location are restored to "c:\temp\".
Restore a uniquely defined backup file from a unique location to a unique location.	LPS_Sync /f "c:\temp\ABCInc062606.lpz" /r "c:\newloc\"	The contents of c:\temp\ABCInc062606.lpz are restored to the c:\newloc\ folder.
Restore from default path and file name – silent mode	LPS_Sync /r /s	The result of this command is similar to the first scenario above. The difference is that no on screen messages will be displayed.
Safe restore from default path; registry files not restored.	LPS_Sync /u	Most recent backup is restored to Software Labeling\backups \safe restore; a file called registry_restore.bat is created in the same folder. You will have to run this file to restore your registry files.

If the LPZ file to restore is not specified and there is more than one LPZ file (in the default or in the specified path), the most recent backup file will be restored.

## Running Backup / Restore on a Schedule

You can run LPS\_SYNC on a scheduled basis. For example, you can schedule a recurring backup on Server A and then restore the backup file to Server B also on a schedule to keep the two systems in "sync." There are different ways of doing this, two of which are demonstrated below:

### Use the Scheduled Tasks option within Windows and lps\_sync.exe

1. Schedule a recurring Backup on Server A.  
An example command requiring no user interaction: LPS\_SYNC.EXE /b /l /y
2. Schedule a recurring Restore on Server B that "points" to the backup file on Server A.  
An example command: LPS\_SYNC.EXE /r /f'X:' /l /y (where X: is a mapped drive to the backups folder on Server A)

### Use the Windows AT command from the command line

1. Schedule a recurring Backup on Server A.

**For Example**

**32-bit:** AT 23:00 /every:M,W,F C:\Program Files\Loftware Labeling\lps\_sync.exe /b /l /y

**64-bit:** AT 23:00 /every:M,W,F C:\Program Files(x86)\Loftware Labeling\lps\_sync.exe /b /l /y

**Note:** The above example would perform an lps\_sync Backup command every Monday, Wednesday and Friday at 11:00 PM, creates a log and requires no user input.

2. Schedule a recurring Restore on Server B that points to the backup file on Server A.

#### For Example

**32-bit:** AT \\<backup computername> 02:00 /every:T,Th,S C:\Program Files\Loftware Labeling\lps\_sync.exe /r /f"X:\" /l /y

**64-bit:** AT \\<backup computername> 02:00 /every:T,Th,S C:\Program File(x86)\Loftware Labeling\lps\_sync.exe /r /f"X:\" /l /y

**Note:** The above command performs an lps\_sync Restore command every Tuesday, Thursday and Saturday at 02:00 AM, creates a log and requires no user input.

The previous examples show that the LPS configurations of two servers can be synchronized on a scheduled basis based on your requirements. One usage would be to back up the production server and restore to the test server, in essence, synching the configurations as often as desired.

#### Related Information

Consult the *Cluster Installation* section of this guide and our Web site information on more sophisticated methods of fault tolerant setups including load leveling, automatic synchronization, and automatic failover.

With Loftware's Internet Printing applications, a company can maintain centralized control over label printing needs while sending or receiving print requests from Client-side computers with attached printers to the LPS via the Internet to any of their satellite offices, production warehouses, anywhere in the world. For example, with Loftware's Internet tools, an assembly plant in Taiwan can connect with the home office in Chicago at any time day or night and print the labels needed for immediate shipment.

Loftware's Internet Printing technology consists of these tools:

### Loftware's WebPush / Web Listener

This technology allows one server-side application to control label printers anywhere in the world. The Loftware Web Push sends print streams across the Internet to a Web Listener application. The Web Listener is a client-side application that receives print streams from the Loftware Print Server (LPS) across the Internet and prints to locally configured or TCP/IP connected Printers, with no client site intervention needed. All the label requests are triggered from a server side application. Labels are sent to vendor sites or satellite offices across the Internet, without the need for expensive WAN connections.

### Loftware's WebClient

A thin On-Demand Print client that initiates a request to generate the label. The Web Client connects to the Loftware Print Server (LPS) and prints across the Internet. The difference between Web Push technology and Web Client is that with the Web Client, the print request is triggered from the client site.

### Internet ActiveX Control

Allows businesses with their own applications to interface with the LPS and print to Client-side printers across the Internet. This utilizes an ActiveX interface across the Internet to print labels to any client-side printer.

#### Related Information

For information about the ActiveX Internet Control, refer to *Internet ActiveX* in this guide. Read this section before proceeding to use iX.

## Internet Printing Considerations

### Performance Considerations

The Performance Considerations section of Chapter 1 (Loftware Print Server) contains a great deal of information about things to consider when developing the type of label and barcode printing solution that gives you the most benefit. Please review that section before continuing with the installation of the



WebClient or the Web Listener. In addition to the information regarding hardware, number of labels printed, label content and Internet considerations like those listed below must be taken into consideration:

- Graphics
- TrueType Fonts
- Internet connection speed
- The type of Web Server
- Speed of the client computer
- How fast the client-side printer can image the data
- How many clients and printers are being driven simultaneously

If you have a serious load on the LPS already, adding more load with 500 WebClients or 500 printers connected to one Web Listener may not be a very workable situation.

Only one LPS can be placed on a single Web Server; however, two Web Servers could each have an LPS, and the load could be balanced between the two. Software compresses the data streams to maximize throughput, but we cannot predict exactly how quickly labels will print using the WebClient or using the Web Push/Listener Application.

With the Web Client, this time is further decreased after the first time the label has been downloaded, as it is cached on the client-side computer for future reference. With fast hardware and Internet connections, labels can be printed in a number of locations with great speed.

**Note:** Software has a web server that you can try out. Send us a test label, or contact Software Technical Support or Software Sales to see how to do that.

## Licensing Considerations

The Internet Printing applications are licensed similar to the way that other clients are licensed, which is by seats. The LPS Premier comes with all the Internet Printing modules. More seats may be added on the fly by calling your sales representative or Software Customer Service and adding seats. Please remember that once a client computer has logged onto the LPS using one of the clients, that seat is considered taken, and even though that client may be offline, that seat is not available to use by any other client computer. If the LPS is stopped, the licenses are cleared, and a different computer may use a client seat at that time. See the [Software website](#) for more specific information on licensing.

## The WebClient

The WebClient application is a web-enabled full-featured on-demand printing application allowing users to access the Software Print Server over the Internet. Detailed, customized operator input screens are created in Software's Design Module and presented to the users in the same fashion as On-Demand Print and On-Demand Print Client.

Please note that a browser is not needed to connect to the Internet. You must install a small application on the client computers. . The following figure illustrates the concept of the WebClient.

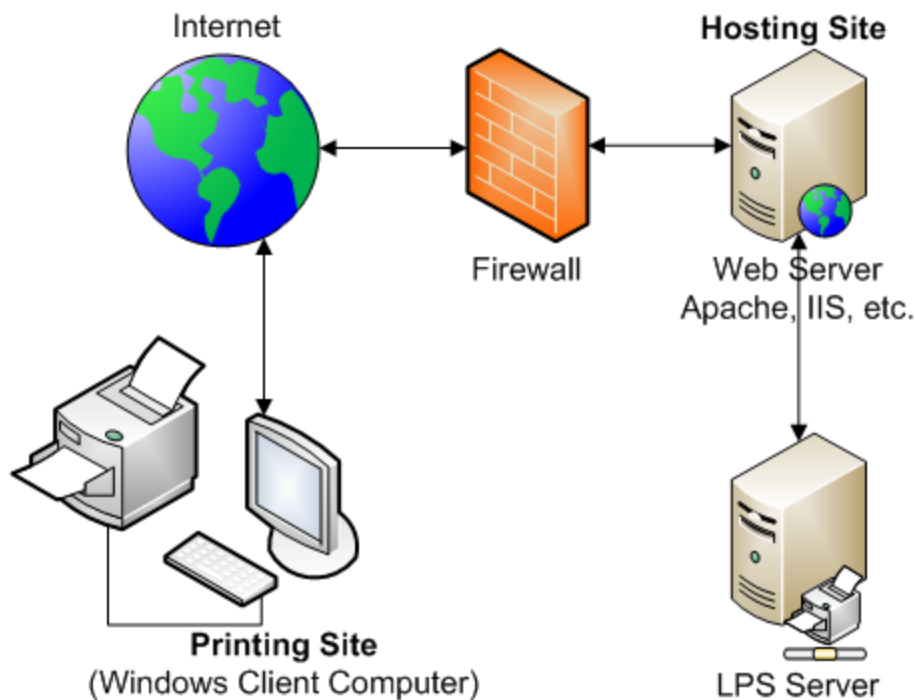


Figure 8.1: The WebClient Design

The Software Print Server and the Web Server are shown as separate entities. They are separate to ensure that print speed is not compromised due a heavy load on the Web Server. This is the recommended setup. The firewall shows that communication through the firewall is possible for companies that have this protection on their network. The data going from LPS to the Clients is compressed (sent as a print stream). Also depicted in this figure are a Windows computer, printer, and Internet connection that are all needed on the Client side to print.

## How the WebClient Works

Similar to Software's On-Demand Print Client, the WebClient does not need the Software engine installed on the workstation computers that are printing labels. This saves costs, installation time, maintenance and designing/configuring the same labels repeatedly on many computers. The WebClient provides users of the LPS system the ability to select and print labels from anywhere in the world via a low cost Internet connection.

**Note:** The WebClient application is licensed by "seats," not "concurrency." This means that the first workstations to log in are recorded as a license taken by the LPS. Once the maximum number of licenses is used, no other WebClient modules are able to execute, even if one logs out. If you want different workstations to be able to use client modules, you must stop the LPS Service and then restart it. This clears the license list kept by the LPS. For complete information on licensing for all Software Products, refer to the Software Web site.

## Choosing the WebClient or On-Demand Print Client

When choosing which application to implement, consider the following: The On-Demand Print Client runs on a LAN or WAN and does not need a web server. On the other hand, the WebClient runs across the Internet, but a web server is required to run it. The following table lists the requirements needed for On-Demand Print (both traditional and client) and the WebClient.

Printing Tool	Requires LAN/WAN	Requires LPS	Requires Internet	Stand-Alone	Requires Web Server
On-Demand Print	No	No	No	Yes	No
On-Demand Print Client	Yes	Yes	No	No	No
WebClient	No	Yes	Yes	No	Yes

## Components of the WebClient

The following figure shows conceptually the relationship among a web server, servlet engine, and web servlet.

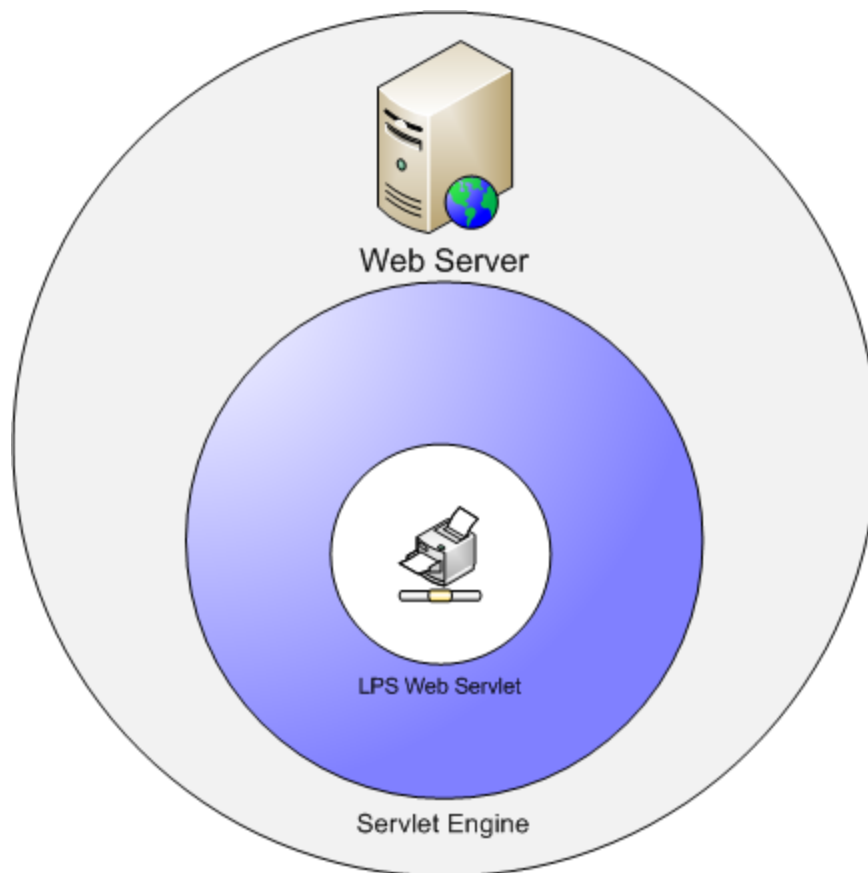


Figure 8.2: Web Server and Servlet Relationship

The following components are needed to run the WebClient:

## Web Server

A Web Server is a computer that delivers (serves up) web pages. Every Web server has an IP Address and possibly a domain name. For example, if you enter the URL `http://www.loftware.com/index.cfm` in your browser, this sends a request to the server whose domain name is `loftware.com`. The server then fetches the page named `index.cfm` and sends it to your browser.

- Any computer can be turned into a Web server by installing server software and connecting the computer to the Internet. The Web Server must be up and running prior to any installation of servlets.

**Note:** Loftware cannot help you set up or troubleshoot your Web Server.

- You may set up restricted access (security) to your Web Server, which requires a User Name and Password when starting Loftware's Web Client. The Web Server secures or protects a resource, like the Loftware URI.
- Your Web Server directs the action of hundreds of clients wherever they may be, in the next room or on the next continent.

## Servlet Engine

Servlet Engines take one of these forms:

- Stand-alone – A standalone engine is a server that includes built-in support for servlets. These work well for initial tasking but usually lack the power of a dedicated web server.
- Embeddable – An embeddable engine is a lightweight servlet deployment platform that can be embedded in another application.
- Add-on – An add-on servlet engine functions as a plug-in to an existing server. It adds servlet support to a server that was not originally designed with servlets in mind. For many companies (including Loftware) that already have servers, this type of servlet engine is often a good choice. The add-on servlet engines utilized by Loftware in the development of the WebClient are Apache Tomcat and Allaire's JRun.

Loftware does not recommend nor endorse any particular servlet engine, just as we do not endorse any printers. There are many servlet engines to choose from, and your selection should be based on what engine serves your company's needs best.

**Note:** When setting up the JSP/Servlet container, make sure the Java VM path is set to Java Version JDK 1.3 or higher. Do not use the JRE; use the JDK/SDK, which can be downloaded from Sun's Web site.

## LPS Web Servlet

The LPS Web Servlet is found in the LPS Premier Edition. The LPS Web Servlet is Java-based web application that enables data communications from the WebClient to the Loftware Print Server through the Internet. The LPS Web Servlet Application has two main functions:

- The first main function consists of a Java Servlet that facilitates the communications between the WebClient and the LPS. Servlets are powerful, portable, efficient, and have great endurance. Since it is written in Java, the LPS Web Servlet can go from being deployed on a Windows computer to a high-end Unix server.

URL access, multi-threading, data compression, and database connectivity are all part of this package. Using the Internet as the conduit, when the WebClient is opened, the URL configuration (address) lets the WebClient know what server to go to. The WebClient establishes a virtual connection with the Software Print Server using the LPS Web Servlet as a mediator. Transactions are marshaled by the WebClient throughout the [Web] Client / Server [LPS] interaction. The LPS gathers the requested data and sends it back to the LPS Web Servlet, and the Web Servlet then sends the data back to the WebClient. Transmissions take place as fast as your Internet connection allows.

- The second main function is administrative. It is displayed through a JSP page used in a browser such as Internet Explorer or Netscape and is an interface to the LPS Web Servlet for troubleshooting and diagnostic purposes. The page contains critical configuration data about the servlet and the LPS. The LPS Web Servlet has been packaged as a Web Archive file (.WAR, in this case, `software.war`)

A Web Archive file is a zip (jar) file that takes the form of a standard file archive format for the Java Platform for deployment to a Java-enabled web server. War files are compatible with all JSP containers that comply with Version 1.1 of the JSP specification.

**Note:** The LPS Web Servlet application does not have to run in a Windows environment; it can run on Linux or a Unix box without Windows.

## Software Print Server

The LPS forms the basis of Software's server-centric approach to barcode labeling systems. Server-Centric means that all barcode label printing in an area, building or enterprise is controlled from a centralized computer on the network.

- The LPS is capable of printing labels from your ERP/MRP II and/or WMS systems, regardless of the platform on which they reside.
- Host computers and operating systems, such as UNIX, AS/400, HP 9000, DEC VAX and Risc/6000 can also request labels through the LPS. It is also a viable solution for client/server computer and RF applications.
- The WebClient is used in conjunction with the LPS, gathering label information, such as data, layout, design, etc. Software suggests that you install the LPS to a different server than the one on which the Web Server resides, so if either of these applications develops a problem, the other component is not affected.

### Related Information

For more information, refer to the *Performance Considerations* section of [software.com](http://software.com) or in *The Software Print Server* section of this guide.

For a comparison of web server software, see Wikipedia under "[External Links](#)" on page 283.

## Installing the WebClient Components

If you have previous experience installing Web Servers and/or Web Servlets, the following steps may seem fairly straightforward and easy to accomplish. However, if you are new to this, please take some time to read this section and the preceding two sections carefully. Also take time to read the information on the Web sites that are mentioned in this section to give you a clear idea of the level of complexity of this task.

As you continue reading this section, complete each step, checking them off as you go:

- Step 1: Install LPS (Software Print Server)
- Step 2: Configure CLIENT DEFINED Printers
- Step 3: Install Web Server
- Step 4: Install Servlet Engine
- Step 5: Install LPS Web Servlet
- Step 6: Install WebClient
- Step 7: Test Connection to WebClient

### Step 1 - Install the Software Print Server

If you have not already done so, please install the Software Print Server (LPS) as outlined in the *Software Print Server and Label Manager Installation Guide*.

### Step 2 - Configure CLIENT DEFINED Printers

CLIENT DEFINED printers are printers configured for future use by a client computer. They must be configured on the server, with pre-set Printer-Specific Options, (PSOs) and Label-Specific Options (LSOs). Refer to the Software Label Manager User's Guide for more information on CLIENT DEFINED printers.

### Step 3 - Install a Web Server

If you do not already have a Web Server, please install one and get it up and running. Software Technical Support cannot help you with the installation of your Web Server or with any problems related to your Web Server.

### Step 4 - Install the Servlet Engine

If you have not already done so, please install a Servlet Engine at this time.

**Note:** Software cannot help you set up or troubleshoot your Servlet Engine.

### Step 5 - Install the LPS Web Servlet into the Servlet Engine

It has been stated previously that you may use any Servlet Engine you would like. In this section, Software includes examples of installation of the LPS Web Servlet on two Servlet Engines, "Tomcat" and

“JRun 3.0”. It is very important that you understand your Operating System before proceeding with the installation of the LPS Web Servlet. In addition, please read Tomcat or JRun documentation, especially the troubleshooting sections.

### Windows Installation of LPS Web Servlet into Tomcat

1. Copy the `loftware.war` file to the directory where you installed Tomcat, for example, `C:\Tomcat\webapps` directory. The `loftware.war` file is found on the Software CD, under the **Servlets** folder.
2. Start Tomcat. This can be done using `startup.bat` located in Tomcat's `bin` directory.
3. After Tomcat comes up fully, shut it down. Tomcat should automatically extract the `loftware.war` file into the WAR file structure under the `Webapps` directory.
4. Modify the `web.xml` file as outlined in the next section. (This sets the address of the LPS Server.)
5. Restart Tomcat for the changes to take effect.
6. Verify the Web Application is properly installed by navigating to the following URL and verifying the LPS Web Servlet page is displayed.

```
http:// <ipaddress:port (if needed -default 8080)/loftware/ LPSRPT.jsp
```

This completes the Windows installation of Tomcat and the LPS Web Servlet application. Tomcat should now be configured to a Web Server such as IIS or Apache using the appropriate adapter or connector. Refer to your Web Server's and your Servlet Engine's (such as Tomcat) documentation.

To reinstall the Web Application, go to the `C:\Tomcat\webapps` directory, delete both the `loftware.war` file and the `Software` directory and re-install as outlined previously.

### Unix Installation of LPS Web Servlet into Tomcat

1. Copy the `loftware.war` file to the directory where you installed Tomcat, for example:  

```
/usr/lib/-tomcat.
```
2. Start Tomcat. After Tomcat comes up fully, shut it down. Tomcat should automatically extract the `loftware.war` file into the WAR file structure under the `Webapps` directory.
3. Modify the `web.xml` file as outlined in the next section.
4. Restart Tomcat for the changes to take effect.
5. Verify the Web Application is properly installed by navigating to the following address and verifying the LPS Web Servlet page is displayed.

```
http:// <ipaddress:port (if needed -default 8080)/loftware/ LPSRPT.jsp
```

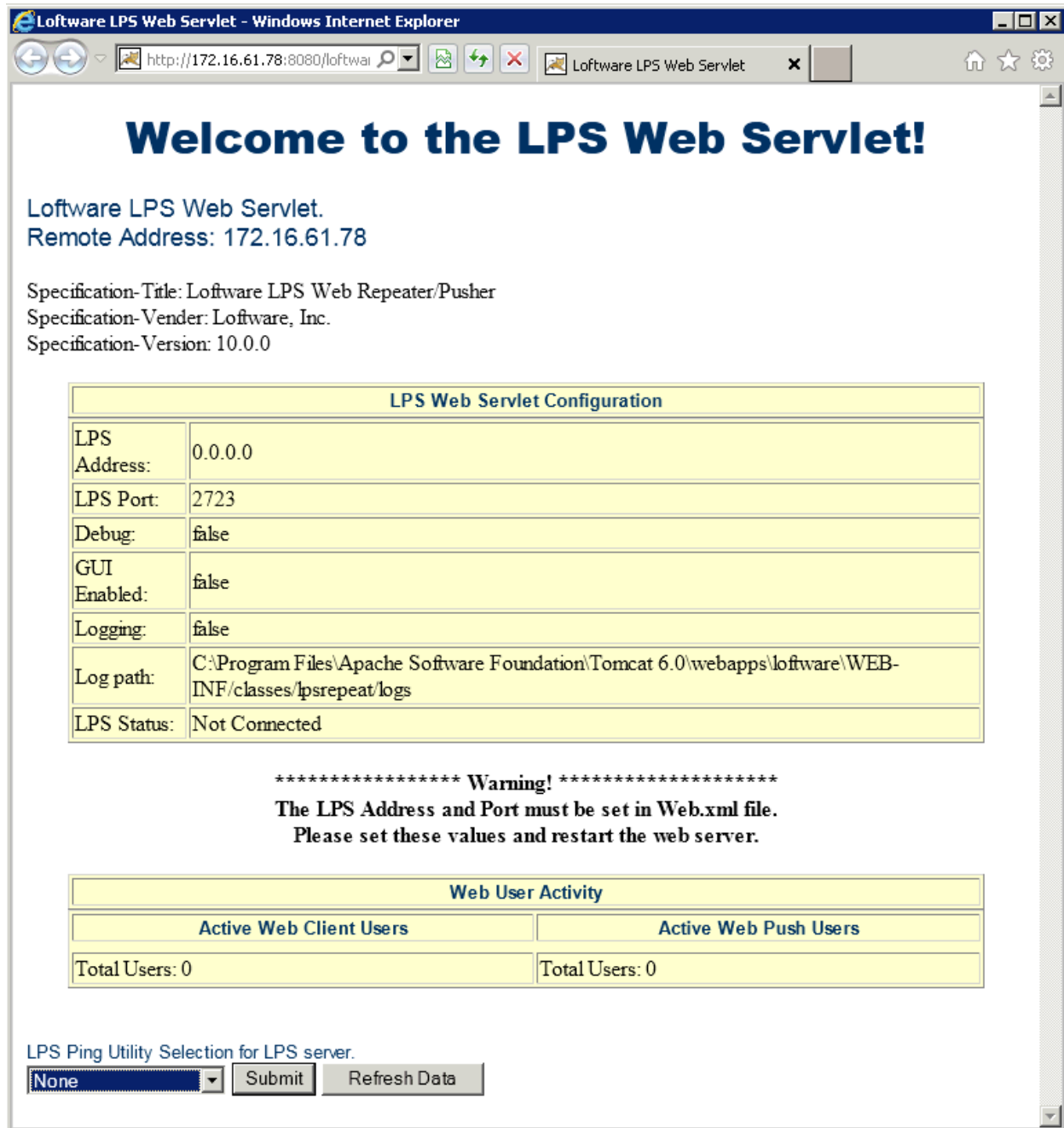


Figure 8.3: LPS Web Servlet Page

This completes the Unix installation of Tomcat and the LPS Web Servlet application. Tomcat should now be configured to a Web Server such as IIS or Apache using the appropriate adapter or connector. Refer to the documentation included with your Web Server and your Servlet Engine.

To re-install the Web Application, go to the C:\Tomcat\webapps directory, delete both the loftware.war file and the Loftware directory, and re-install as outlined previously.



## Windows Installation of LPS Web Servlet into JRun

1. Open Jrun on your Server.
2. Click the JRun default server in the left pane of JRun's JMC.
3. Click **War deployment** in the main (right side) pane
4. Enter the following properties in the right pane:
  - a. Servlet War File or Directory: Path to software.war, such as C:\<Program Files Folder>\Software Labeling\software.war. The software.war file is also found on the Software CD.
  - b. JRun Server Name: Select Installation Server, that is, Jrun Default Server
  - c. Application Name: software
  - d. Application Host: All Hosts
  - e. Application URL: /software
  - f. Application Deploy Directory: C:\JRun\servers\default\software
5. Click **Deploy**. If unsuccessful, fix any errors and try again.
6. Modify the web.xml file as outlined in the Web.xml file modification section. This sets the address of the LPS Server.
7. Restart the JRun server.

Verify the Web Application was properly installed by loading up a browser and go to `http://ipaddress:port (if needed –default 8100)/software/ LPSRPT.jsp` and verifying the LPS Web Servlet page is displayed.

This completes the installation of LPS Web Servlet Web Application on JRun. JRun should now be configured to a Web Server such as IIS or Apache using the appropriate adapter or connector. Refer to documentation for your Web Server and your Servlet Engine (such as JRun).

**Note:** The WebClient only works if you have the LPS configured. See *Understanding the Software Architecture* section of this guide for information on installing and configuring the LPS.

**Note:** Be sure to save the changes and restart the JSP container, (For example, Tomcat or JRun).

To re-install the LPS Web Servlet Application, select the JRun Default Server/Web Applications/LPS Web Servlet in the left pane of the JRun Management Console. In the main pane, click **Delete** to remove the LPS Web Servlet Application, then re-install the Web Application as outlined previously.

**Note:** After installing and upon re-starting the servlet engine, a directory structure is created. The web.xml file that is created upon the re-start must be modified. See following instructions:

### Web.xml file modification

The web.xml file is found in the newly created directory structure, which is seen in the figure below. Your files may look slightly different, depending on your OS and which servlet engine you have

installed.

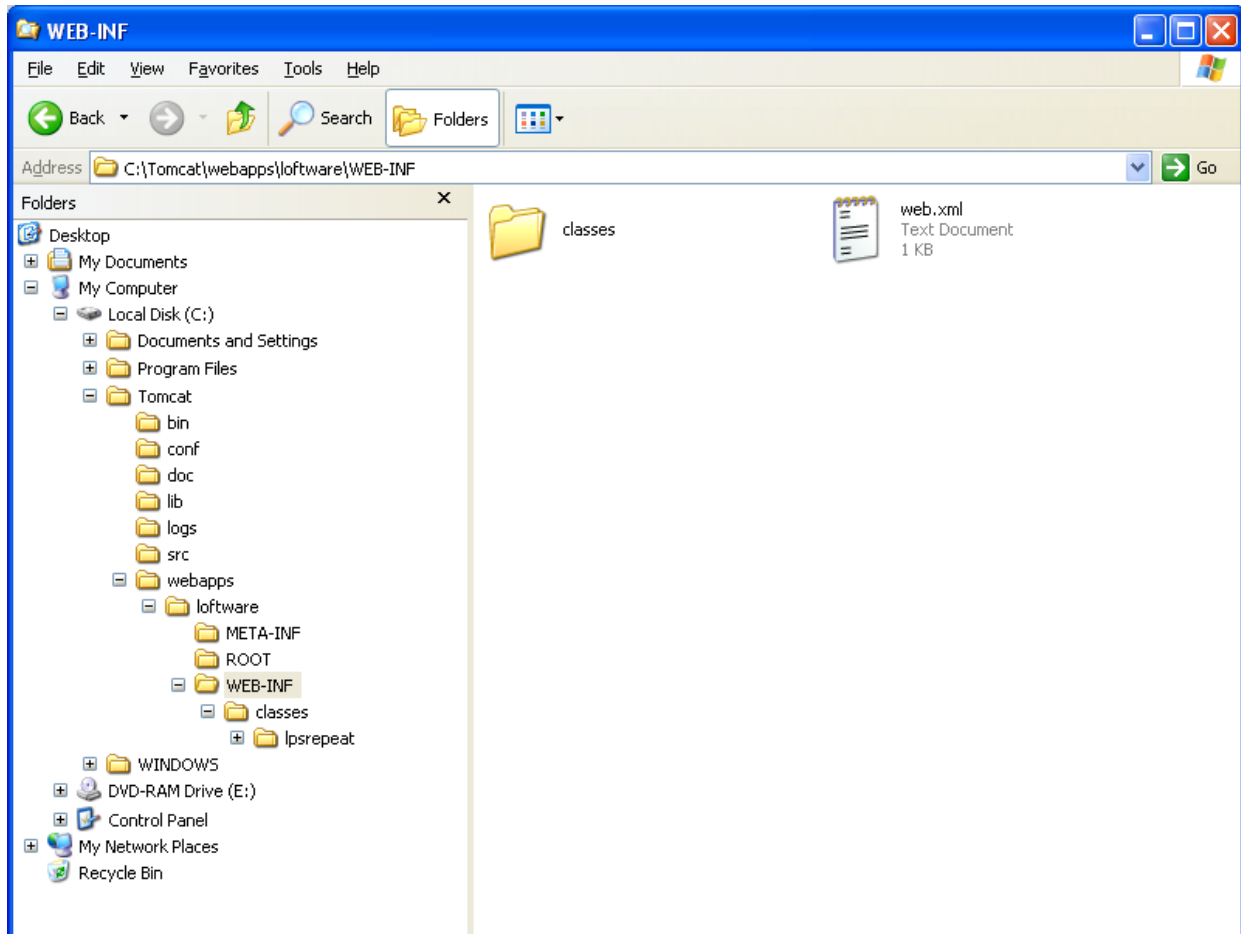


Figure 8.4: Typical directory structure of servlet engine after installing in Windows.

After installation of the Software Web Application, you must modify the servlet configuration file (web.xml) so that the application is able to locate the LPS Server.

In order for the changes to take effect, the servlet engine must be restarted.

### The WAR file

The software.war file has a descriptor file named web.xml that is located in the WEB-INF directory. This file is an XML file that is used to configure the Software LPS Web Servlet Application. The web.xml file contains a section called SOFTWARE Servlet Configuration, in which only one area needs to be configured, that being the IP Address to the LPS.

Configuration Areas	Property
LPS IP Address	Example: 172.17.0.137

#### Example web.xml

```
<!-- SOFTWARE Servlet Configuration -->
```

```
<context-param>
  <param-name>LPSAddress</param-name>
  <param-value>180.10.0.231</param-value>
```

**Note:** Change only the values in between the > and < tags (180.10.0.231).

```
<description>
  Ip Address of the LPS Server.
</description>
</context-param>
```

## Step 6: Install the WebClient

Install the LPS WebClient to the Client computer using one of the following ways:

- A Full install from the CD adds the WebClient executable.
- An install from the LPS Clients folder or LPS WebClient folder on the CD.
- A WebClient install hosted on the Software Print Server and downloaded by the Client computer.

## Step 7: Test the Connection to the WebClient

1. Design the necessary label formats (using Software Label Manager), making sure that you customize your operator input screen.
2. Configure the printers for these labels as CLIENT DEFINED in Software Label Manager and put them on the server.

**Important:** All configured printers to be used with the WebClient must be configured as "CLIENT DEFINED," meaning that they are manipulated by the WebClient and not by the Software Print Server.

3. Launch the Client from a shortcut on the desktop or Start | Program Files | Software Labeling | Print Server | Internet Clients | Web Printing Client.

The first time you open the WebClient on the Client computer, the following window appears.

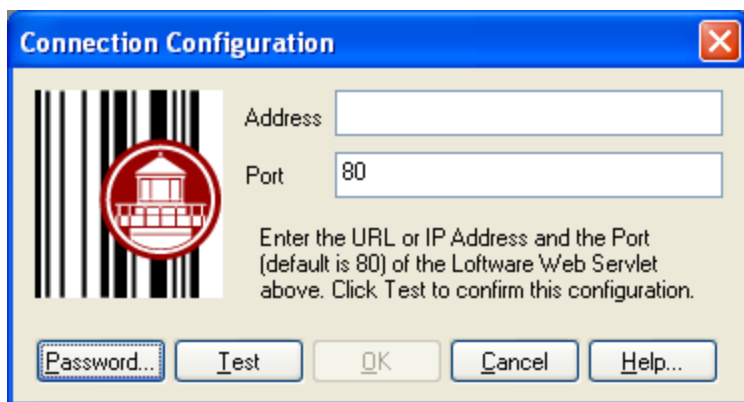


Figure 8.5: Configuring Web Server Connection

1. Enter the IP Address and Port or URL and Port for the Web Server to which you are connecting.

The IP address may be 159.64.0.149, while a URL may be www.example.com.

2. Click **Password** if the administrator has set up security on your Web Server, and enter the authentication information.

**Note:** If a password is required, and Test is pressed first rather than Password, a message box is displayed prompting you for your user name and password, and these must be entered.

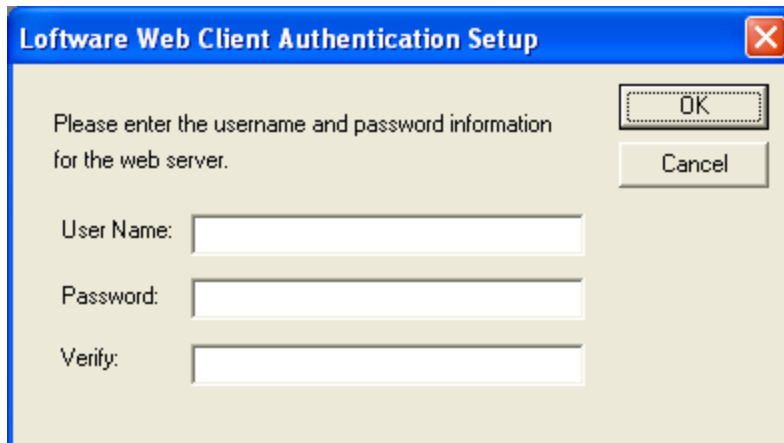


Figure 8.6: Web Client Authentication

3. Enter the Authentication Information, click **OK**, then click **Test**.

**Note:** The test function is required to ensure that the correct Address and Port have been entered. The OK button is not enabled until this function has been performed successfully.

4. If the test is not successful, check the error message from the Web Server (example shown in figure below), click **OK**, correct the error, and try again.

**Remember:** Software cannot help you with Internet connections and address problems.



Figure 8.7: Connection Error from Web Server

5. If the test is successful, a "Successfully Connected" message is displayed, press OK to close the box.

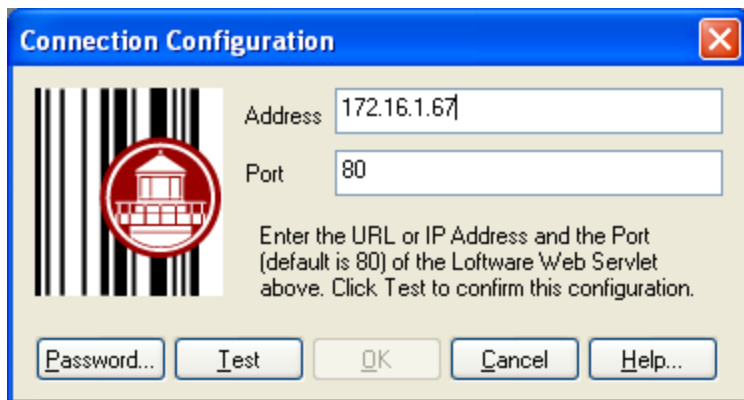


Figure 8.8: Connection Configuration

6. Click **OK** again to open the **File Open** dialog box on the server.

The first time it is opened, the Software WebClient File Open dialog box is displayed within the WebClient screen. This dialog box contains the list of labels and/or folders that are on the server. This list of labels and/or folders is the same as the root directory of the Labels Path on the LPS.

After the first time WebClient is run, the last label opened is displayed, as the label design and device configuration are stored in a cache file once they are opened.

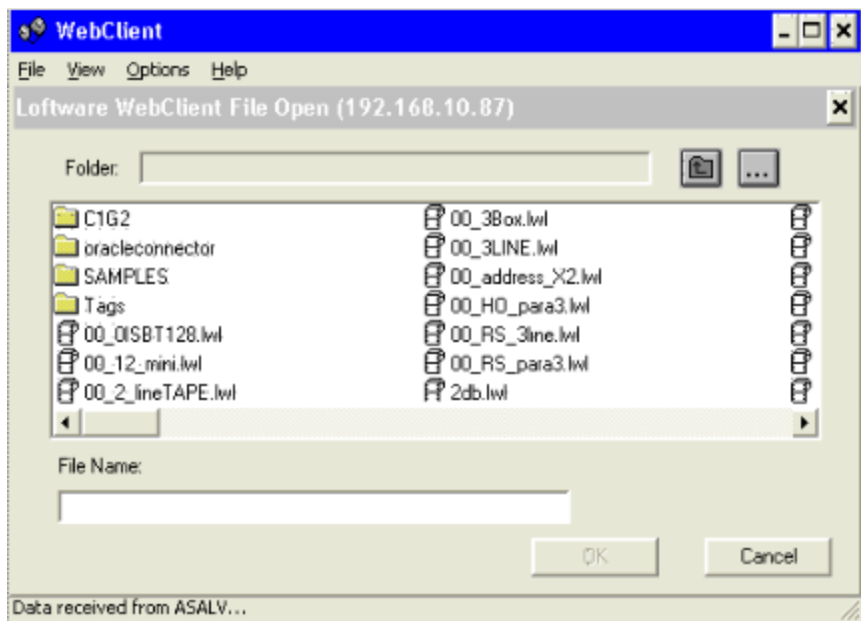


Figure 8.9: The WebClient window, displaying designed/configured labels

7. Choose the Label you want to print from the Labels directory. The first time you open this label, one of the following warning messages appears:



Figure 8.10: Request for CLIENT DEFINED Printer for Label

This message indicates that the printer cannot be locally configured, because it has not been added to the LPS as CLIENT DEFINED. You must configure this printer on the LPS before continuing.



Figure 8.11: Request to configure the label locally

The second message indicates that the printer does exist on the LPS as CLIENT DEFINED, but now must be configured locally.

8. Click **Yes**. The **Configure Device** window appears. The drop-down list displays all the CLIENT DEFINED configured devices on the Server. None of the printers have been locally configured.

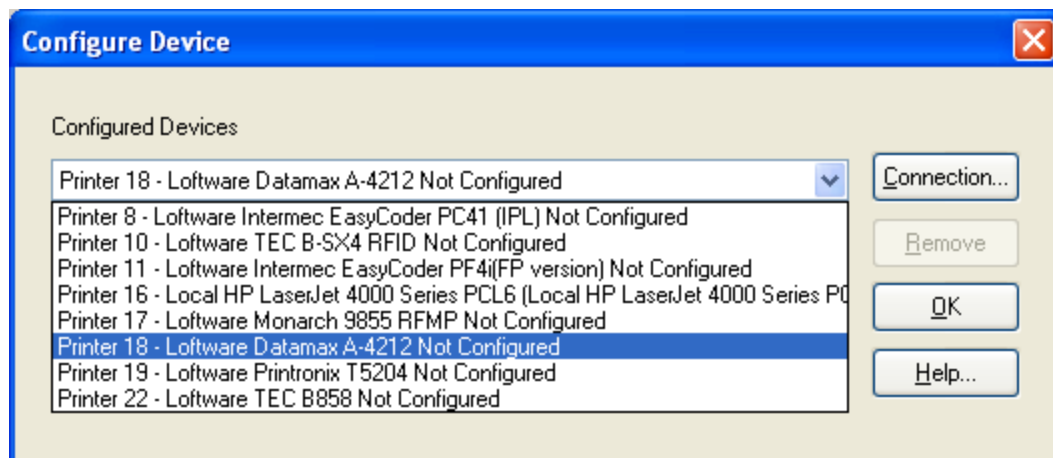


Figure 8.12: Configured Printers List

9. Select the printer the label is designed for; click **Connection**. The following window appears.

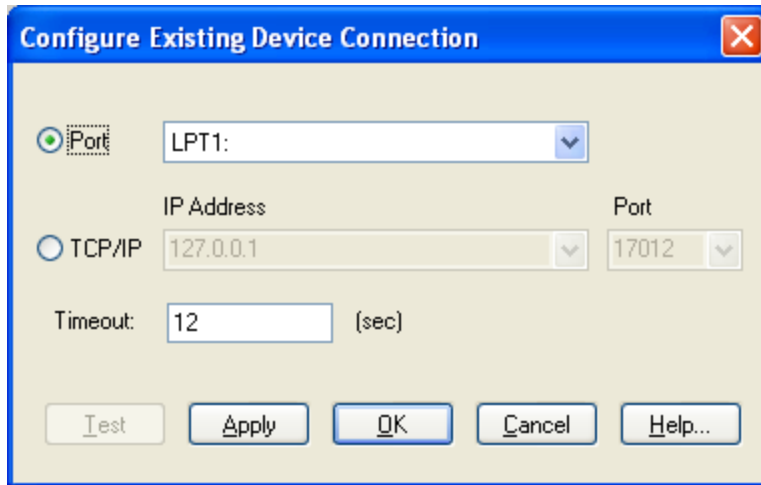


Figure 8.13: Configuring Local Printer Connections

10. Choose the type of connection, Port, or TCP/IP.
  - a. Click **Apply**, then **Test** to send a test print stream to the printer.
  - b. Click **OK**, and **OK** again to close the window and return to the label to begin printing.

**Note:** Test your connection the first time you print a label.

- To change or remove local printer configurations, press F6 or select File | Devices.

**Note:** Remember that in order for a USB Port to be displayed in the Port list, the printer must be connected to the computer and powered on.

#### Related Information

For information on device connections and configurations, refer to *Device Connections* in the *Software Label Manager User's Guide* and *The Software Print Server* section of this guide.

## Using the WebClient

The WebClient window is similar to using the On-Demand Print Client window, as shown below. Note the information displayed in the status bar.

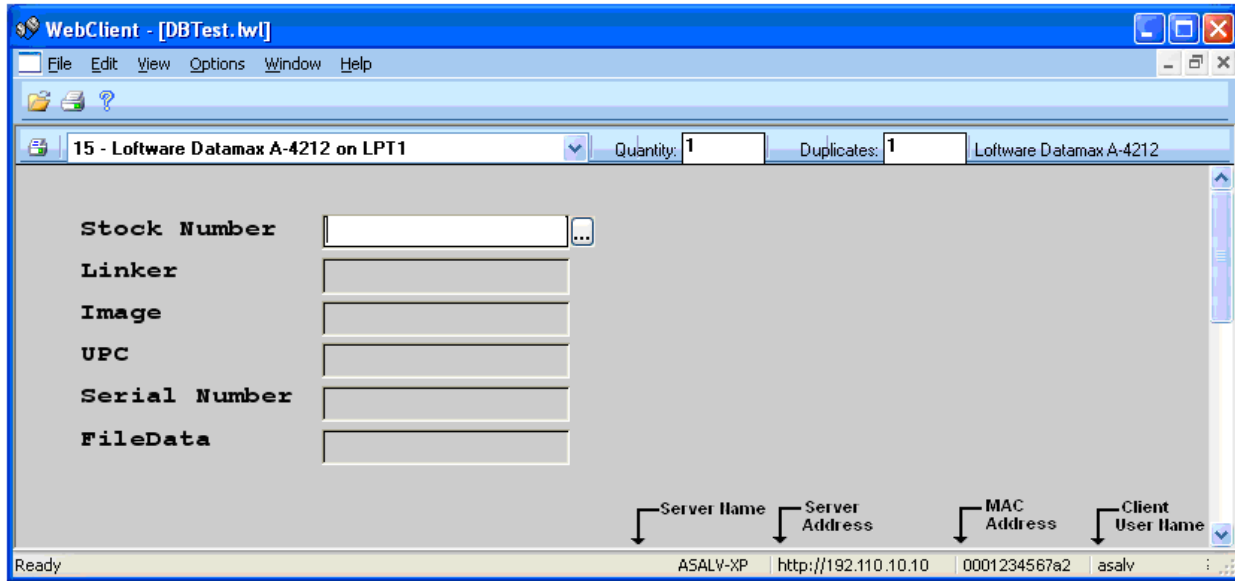



Figure 8.14: The WebClient Screen

Make sure printers are connected and have been tested prior to this operation.

1. Start the Client from a shortcut or **Start**. The Software WebClient window is displayed.
2. Select File | Open. The **Open** dialog box displays the Label File.
3. Double-click the label you want to print from the Label Directory, or select it and click **OK**.
4. Enter information in the data fields manually, or if the label is connected to a database, click the database browse button (...).
5. Change the **Quantity** and **Duplicates** if needed. (Default = 1)
6. Click Print , or select File | Print | OK.

## Additional WebClient Information

### The Cache Directory

A cache directory is created when the first label downloaded from the LPS is opened on the client computer. Labels that are subsequently opened are also saved in this directory. This keeps you from having to download the label formats, layouts and printer information each time this label is opened. This enhances the WebClient performance after the label formats, etc. have first been received from the LPS. The cache directory is created under C:\<Program Files Folder>\Software Labeling, and as you open each label, the directory structure that is created is replicated in the same manner as the label directory of the LPS to which you are connecting.



The Cache Directory may be purged by choosing Options | Purge Cache Directory from the menu bar. You may wish to purge the Cache Directory if you have opened up many labels over a period of time, and they are taking up a lot of disk space on the Client computer. This may be especially useful if you are running the WebClient on a computer with a limited amount of disk space, or if you have opened up a number of labels you are no longer using.

## Troubleshooting the WebClient

### Diagnostics

Open the Diagnostics window by clicking View | Diagnostics from the menu bar of the WebClient window.

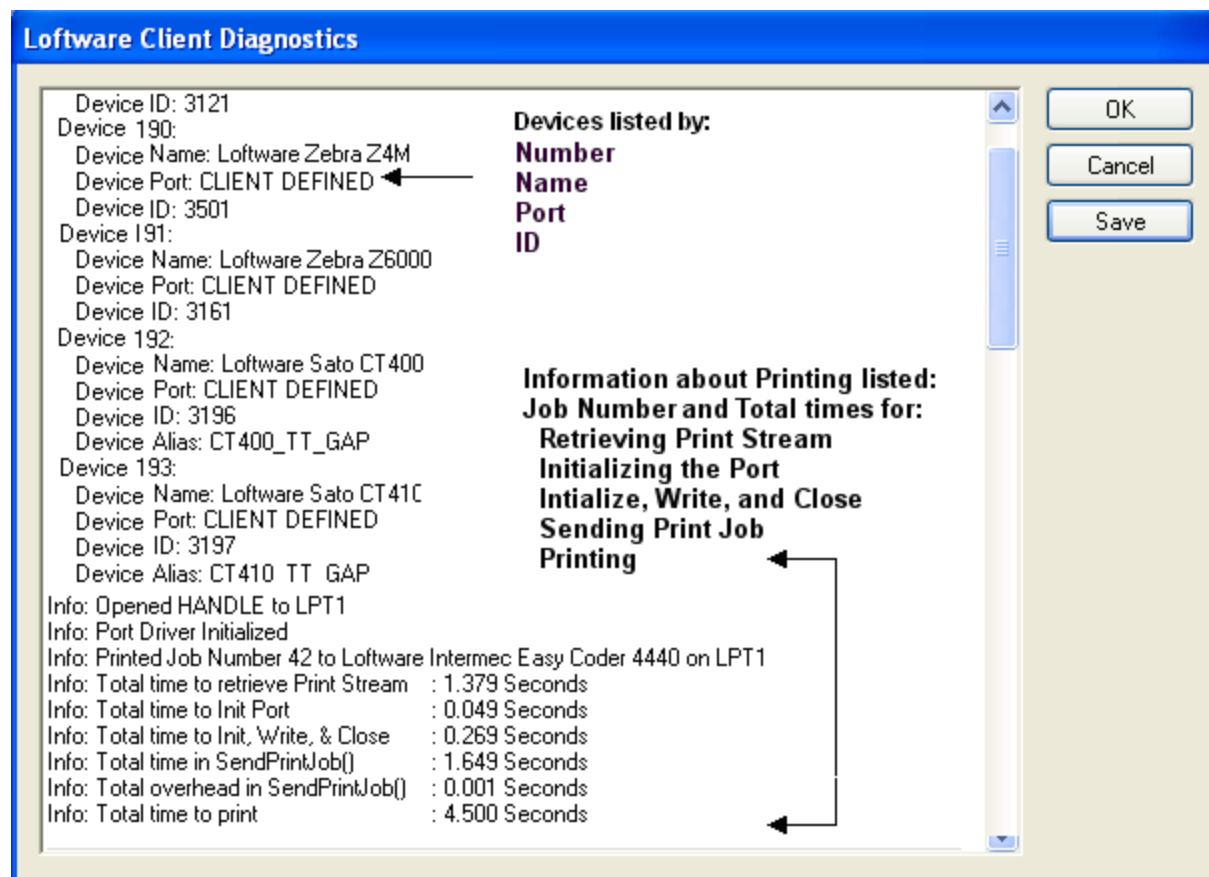


Figure 8.15: Software Client Diagnostics

The Diagnostics window is useful to see any errors that may have occurred with printing, server connections, etc. It displays the total printing time as well, which may be quite useful if you are comparing printing time with a dialup connection versus a cable modem connection. Technical Support may view this file to help you troubleshoot printing and/or connection problems as well. If you are still having trouble printing using the Client On-Demand Printing module, read the following checklist before calling Technical Support.

1. Has your connection been tested?
2. Verify that the right printer has been correctly configured for your labels.  
(Both locally and on the LPS.) If it has not, configure the printer.
3. Verify that all server information is correct, such as the IP address, etc.
4. Verify that you can test print a label to the selected printer.

## Troubleshooting Servlet Engines Tomcat and/or JRun

It is recommended to install Apache Tomcat in the root directory. (On Windows in the C:\Tomcat directory and on Unix in the /usr/lib/tomcat directory.) This makes the paths short and easier to find. We recommend you review the Apache Tomcat documentation. For information, see Apache under ["External Links"](#) on page 283.

- Did you create an environment variable called JAVA\_HOME and set it to the JDK 1.3 location? Example: JAVA\_HOME=C:\jdk1.3
- Did you add %JAVA\_HOME%\bin to your Path so the Java command is recognized from the command line? This must point to the JDK1.3\bin directory.
- Did you create an environment variable called TOMCAT\_HOME and set it to the tomcat directory? Example: TOMCAT\_HOME=C:\Tomcat.
- Did you verify that Tomcat is installed correctly by going to a browser and accessing the default Tomcat JSP page? Example: <http://Address: Port (if needed –default 8080)/examples/jsp>. Click an execute option and make sure the Tomcat example JSP executes. This tests the installation of Tomcat. If this fails, make sure the previously steps are correct.
- Did you install JRun in the root directory? (On Windows in the C:\JRun directory and on Unix in the /usr/lib/JRun directory.) This makes the paths short and easier to find. Also, make sure that JRun points to the JDK1.3\bin directory. This value is set during the JRun installation.

## More Information

JRun documentation can be found at: [Start | JRun 3.0 | JRun documentation](#). Specific information can also be found under *Deploying J2EE Applications Using the JMC and the JRun Quick Product Tour* located in the JRun Management console. Click **App. Deployment** and then **Deploying a Web Application**. Instructions for installing Web Applications are found here. Software recommends that you read these documents.

## Troubleshooting Printing with WebClient

While Printer Configurations are relatively simple when using the WebClient, there are some points to keep in mind to ensure smooth operation of all print jobs. Listed below are some potential issues that may occur when first using the WebClient.

Issue	Solution
The Printer to which you want to print is not showing up in the WebClient Window	Make sure you have selected the printer for which the label was designed, or the drop-down printer list does not display this printer. Remember that this application prints only the label(s) that has been designed for a specific CLIENT DEFINED printer that is configured on the server. You are not able to print the label using another printer when using WebClient on a Workstation computer. If you want the same label to be printed by the WebClient on two different printers, then that label must be created in Software Label Manager for each printer; both printers must be set as CLIENT DEFINED, and must be available on the server.
Labels are not printing, even though the drop-down printer list is populated	Check your printer connections. For example, check to make sure the right stock has been chosen for the CLIENT DEFINED printer.

### Related Information

For information on printer connections, refer to *Device Connections* and *Printers and Labels* in the *Software Label Manager User's Guide*.

## Definition of Terms

Before installing or using the WebClient, the Web Listener, or the Internet ActiveX Control (iX), please take time to familiarize yourself with the following terms.

Term	Definition
Apache	A public domain Web server developed by a loosely knit group of programmers. Because of its sophisticated features, excellent performance, and low price (it is free), Apache has become the world's most popular Web server. The original version of Apache was written for UNIX, but now there are versions that run under Windows and other platforms.
Internet Information Server (IIS)	Internet Information Server, Microsoft's Web server that runs on Windows platforms. Because IIS is tightly integrated with the operating system, it is relatively easy to administer.
Java	A high-level programming language developed by Sun Microsystems. Compiled Java code can run on most computers because Java interpreters and runtime environments, known as Java Virtual Machines (JVMs), exist for most operating systems, including UNIX and Windows. Java is a general purpose programming language with a number of features that make the language well suited for use on the World Wide Web.
Java Virtual Machine (JVM)	A Java interpreter, the Java Virtual Machine is software that converts the Java intermediate language (bytecode) into machine language and executes it. The original JVM came from the JavaSoft division of Sun. Subsequently, other vendors developed their own; for example, the IBM Virtual Machine is IBM's Java interpreter. A JVM is incorporated into a Web browser in order to execute Java applets. A JVM is also installed in a Web server to execute server-side Java programs. A JVM can also be installed in a client computer to run stand-alone Java applications.

Term	Definition
Java Development Kit (JDK)	A Java software development environment from Sun. It includes the JVM, compiler, debugger and other tools for developing Java applets and applications. Each new version of the JDK adds features and enhancements to the language. When Java programs are developed under the new version, the Java interpreter (Java Virtual Machine) that executes them must also be updated to that same version. You must have Java Version JDK 1.3 or higher installed on your Web Server prior to adding the Servlet Engine. [Not the JRE!]. This may be found at <a href="http://java.sun.com/j2se/">http://java.sun.com/j2se/</a>
JavaServer Pages (JSP)	JavaServer Pages, created by Sun, this is a way to write snippets of servlet code directly within a static HTML Page. Blocks of servlet code are called scriptlets and may use one or more of four variables: request, response, out, and in. The file extension for JSPs is .jsp.
Linux	A freely-distributable open-source implementation of UNIX that runs on a number of hardware platforms.
Servlet	A servlet is a generic server extension that can be loaded dynamically to expand the functionality of a web server. Servlets are commonly used with web servers, and run within a Java Virtual Machine (JVM). Since servlets are all handled by separate threads within the web server process, they are very efficient and scalable. Servlets are supported on ALL platforms that support Java, and servlets work with all the major web servers. Loftware has developed a servlet called the LPS Web Servlet for use with the WebClient.
Servlet Engine	Servlet Engines are designed to test and deploy servlets. Your choice of servlet engine depends upon the Web Server you are running. Keep in mind, however, that Loftware has developed the LPS Web Servlet to operate within Java Version JDK 1.3 or higher.
Unix	A popular multi-user, multitasking operating system developed at Bell Labs in the early 1970s. UNIX was designed to be a small, flexible system used exclusively by programmers, and was one of the first operating systems to be written in a high-level programming language, namely C. Historically, it has been less popular in the personal computer market, but the emergence of Linux is revitalizing UNIX across all platforms.
Uniform Resource Identifier (URI)	Uniform Resource Identifier is the generic term for all types of names and addresses that refer to objects on the World Wide Web.
Uniform Resource Locator (URL)	Uniform Resource Locator is the global address of documents and other resources on the World Wide Web. A URL is one kind of URI. The first part of the address indicates what protocol to use, and the second part specifies the IP address or the domain name where the resource is located. For example, the following URLs point to two different files at the domain loftware.com. The first specifies an executable file that should be fetched using the FTP protocol; the second specifies a Web page that should be fetched using the HTTP protocol:  ftp://www.loftware.com/stuff.exe http://www.loftware.com/index.html
Web Server	A computer that delivers (serves) web pages. There are many Web server software applications, including public domain software from NCSA© and Apache©, and commercial packages from Microsoft and others.

## Using the WebClient with a Database

The WebClient is able to print Serial Numbers, Incrementing/Decrementing Fields, Check digits, Formulas, etc.

### Example

If your label has a Database Key Field, use one of the following options to access the database information for your label:

1. Click the browse button (...). The first 25 records in the database are displayed.
2. Refine the search by typing one or more characters in the **Search Key** field, and clicking **Go**.

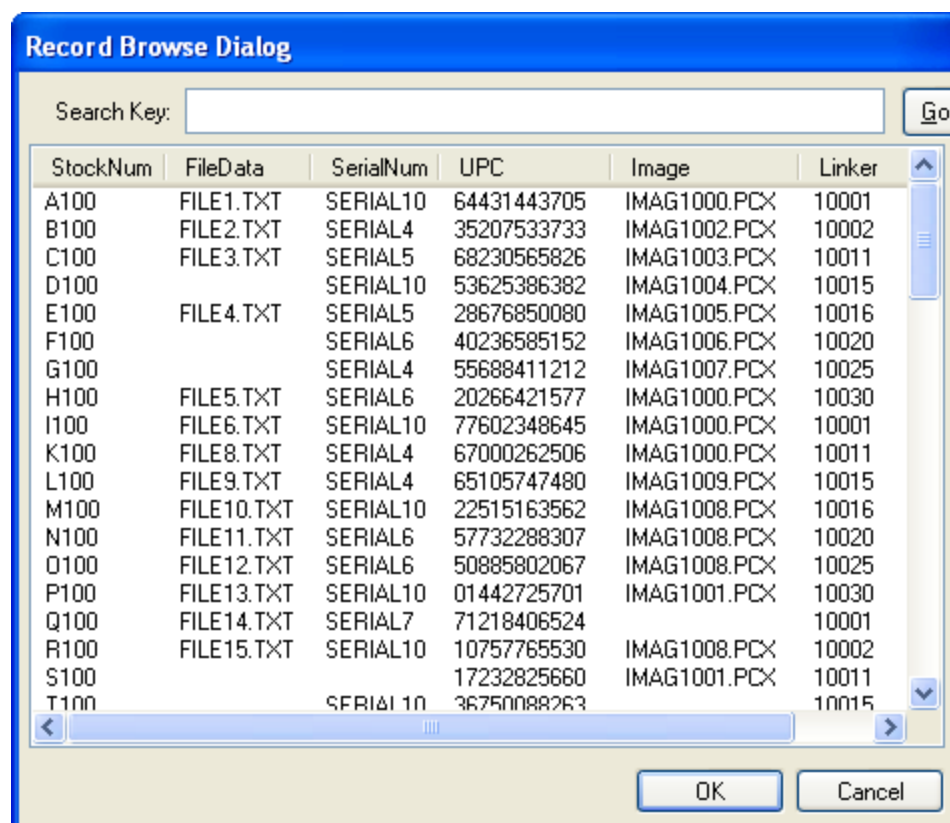


Figure 8.16: Record Browse Dialog Box

Alternatively, to filter the database, enter one or more valid character(s) in the key field and click the browse button (...). A dialog box with option buttons appears.

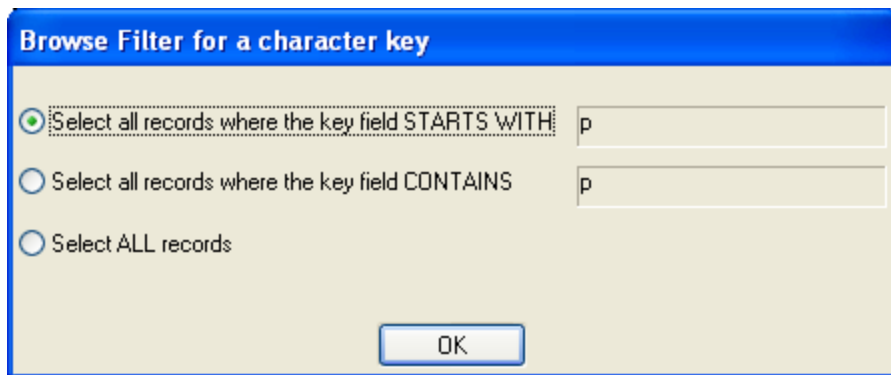


Figure 8.17: Browse Filter

1. Choose one of the three options; the database is filtered to display only database fields that start with the character(s), contains the character(s), or all records.

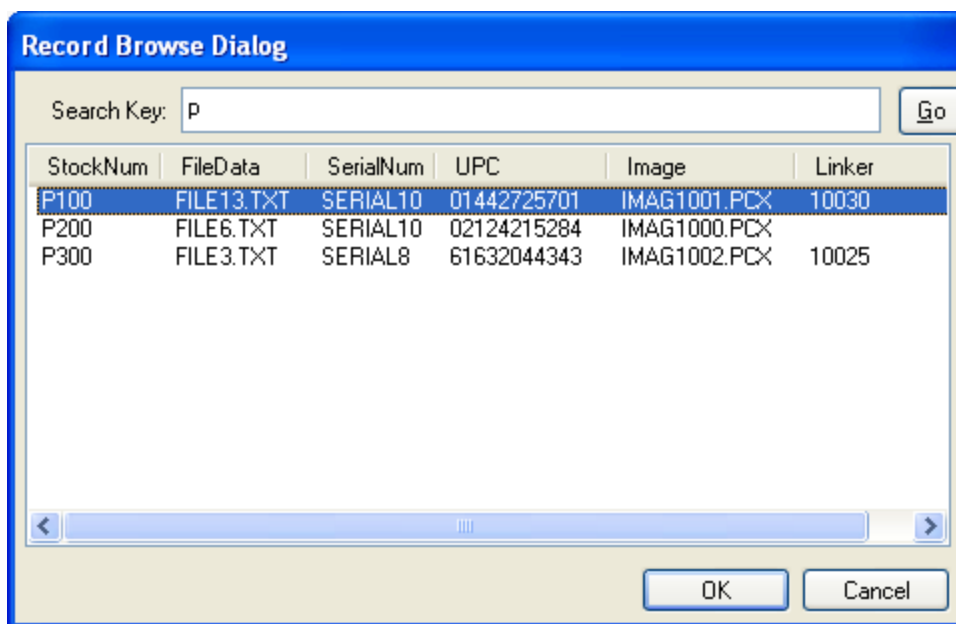


Figure 8.18: Browsing the database using the Search Key

2. Double-click the record you want to open.
3. Print the label(s) by clicking on the Printer Icon, select File | Print | OK from the menu, or press **F9**, and then click **OK**.

#### Related Information

For complete information on Data Sources, refer to the Data Sources section of the *Software Label Manager User's Guide*.

## Internet Data Push and the Web Listener

Data push is the concept of requesting data from one central spot and, in the case of label printing, having it print anywhere on your LAN/WAN or Internet. Software has been performing data push on company LANs/WANs for many years, but Internet data push is relatively new. Using advanced technology, client sites can set up printers and receive print requests from the LPS server across the Internet.

Data Push allows one server-side application to control label printers anywhere in the world. You can “push” labels to your vendor sites or satellite offices without the need for expensive WAN connections. The difference between data push technology and our web client technology is that with the web client, the print request is triggered from the client site. With data push, no client site intervention is needed; all the label requests come from a server-side application.

Software's Internet data push application is called Web Push. The concepts of Web Push and the accompanying Web Listener are visually displayed in the following figure:

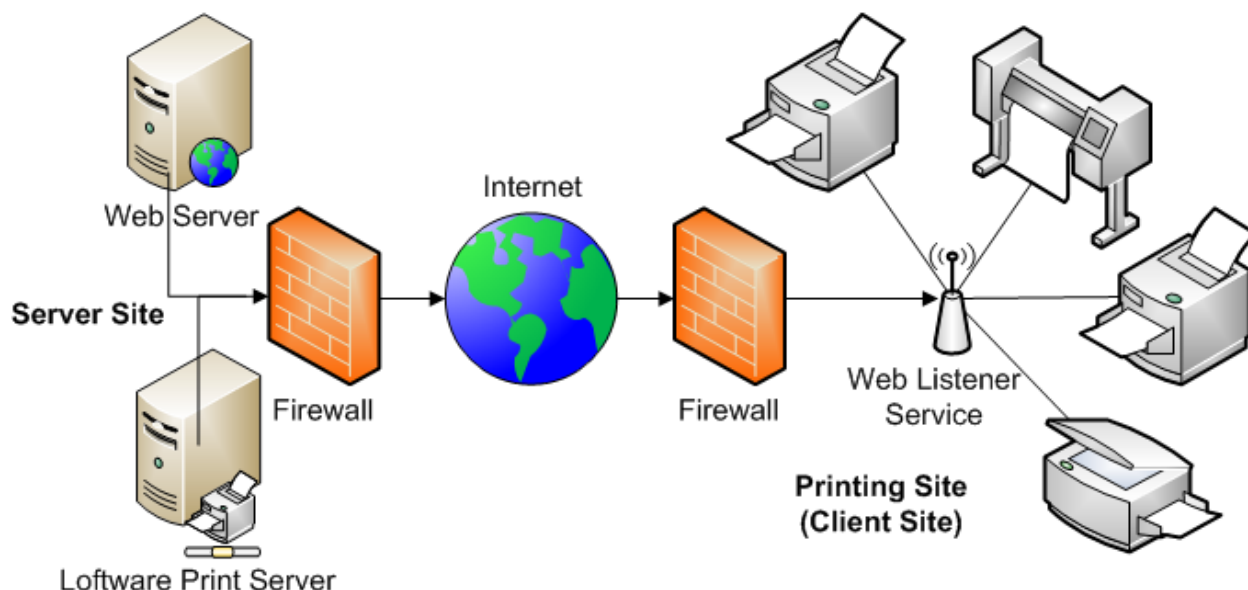


Figure 8.19: About Internet Data Push

**Note:** One Listener can service as many printers as you would like (or have seats for), but to improve throughput, you may want to divide the load, such as one Listener for every 10 printers.

There are three major subparts to the Push system:

### The Software Print Server

With Web Push technology, the Software Print Server does its usual job of receiving and handling print jobs, but in addition, when a job request is received via a .pas, .csv, or .xml file the LPS forwards notification of the job to the LPSPushServlet. In order for the job request from the LPS to be successful,

Internet and printer configuration are required. More information on configuration follows.

## LPS Push servlet

LPS Push servlet is a Java Servlet running on the web server that manages the job notifications coming from the LPS. It is a “middle man” so to speak. When the LPS gets a job request, it sends notification to the LPSPushservlet, and the servlet in turn sends the notification of the job to the Web Listener. The Web Listener in turn gets the data stream from the LPS via the LPS Push servlet and sends an update back (again via the servlet) when the print job is complete.

## Web Listener

This is the Client-side application that connects to the LPSPushservlet in order to receive Print Jobs. Thus, the name “listener” is apt, because this application “listens” for print streams coming from the LPS via the LPSPushservlet. Initial printer configuration is required; more information on this follows.

**Note:** If the Web Listener is not available when a request to print a label is made, the XML file for the job is stored in the \spooled folder. When the Web Listener becomes available, the job will then be printed.

## Installing Web Push / Web Listener Components

**Note:** Web Push and Web Listener are available only in the Software Premier Edition.

### Step 1 - Install the Software Print Server

If you have not already done so, install the LPS.

### Step 2 - Install the Web Server

If you have not already done so, follow the instructions in this guide regarding Web Server installation. Software cannot help you with the installation of your Web Server.

### Step 3 - Install the Software Web Servlet

If you have not already done so, follow the instructions earlier in this section to install the Software Web Servlet.

### Step 4 - Install the Web Listener Client

The Web Listener may be installed one of three ways:

- A Full install adds the Web Listener Client executable when you choose to initialize the Software Print Server during install.
- A separate Client install from the Clients Folder on the CD, found under LPS Internet Installations | InternetPrintingClients.exe.
- An individual Web Listener Client install, either from the CD, or one that is hosted on a Web Server and downloaded to the Client computer, called WebListen.exe.



After completing these steps, you will be ready to configure users and printers and start printing over the Internet.

## Configuring the Web Push Components

Server and Client-Side Configuration - The Printer Configuration information that follows is a two-part process. First, the printers that the client intends to use are configured on the server side, and then the Local or LAN/WAN Ports are configured on the Listener (Client) side.

### Configuring Web Push (Server Side)

The Software Print Server keeps a list of all the Listener Clients that can connect. When a Listener connects, the printer configurations on the Server for that Listener are downloaded to the Listener. Each of these printers must then be configured with a Port on the Listener Side.

#### Step 1: Add a Web Push Client to the Server (LPS) database

1. Open Software Design 32 on the computer running the LPS (Server-side).
2. Open the Device Configuration dialog box: File | Devices or F6
3. Highlight a printer, click **Connection**.
4. Select **Web User** in the **Print Using** section of the configuration dialog box, and make sure that **Shared Network Printing** is checked.

**Configuring Existing Device Connection Settings. (Software Zebra ZM4...)**

**Print Using**

☐ Port: LPT1:

☐ Print Manager/ Spooling: Adobe PDF

☐ TCP/IP: IP Address Port

☒ Web User: [User List] ...

☒ Shared Network Printing

**COM Port Settings**

Baud Rate: 9600 Parity: Odd

Data Bits: 8 Stop Bits: 2

Flow Control: Xon/Xoff

**Advanced Settings**

Timeout: 8 (seconds)

Job Wait: 300 (seconds)

☐ Disable Status Checking ☐ Force Extended Mode

☐ Asynchronous (LPT)

OK Cancel Help

Status

Figure 8.20: Web Push User Configuration

5. Select the designated user from the list. If the user does not exist on this list, click the browse button (...). The LPS Users dialog box is displayed:

Figure 8.21: LPS Users Dialog Box

6. Click **Add**, and then enter data in the **Selected User Data** section. **User Name** is the only required field.

**Note:** We recommend that you use a full name to avoid confusion. The User Name must be unique, meaning that no two people on the User List may have the same User Name, or a warning message is displayed.

7. Click **Apply** to save the addition and create more LPS Users as necessary
8. Click **Add** to add as many LPS users as you wish.
9. Click **OK** to save the addition(s) and exit the dialog box.

## Step 2: Configure Printer(s) for the User

Configure the printers for the Web User in the same manner as you have previously added and configured printers. See Connecting Devices in the *Loftware Label Manager User's Guide* for more information.

## Client-side Connections

### Configuring Connections to the Web Listener

**Note:** The following steps detail your connection and configuring Printers on the Listener (Client) Side.

1. Open Web Listener – Interactive.
2. Click **Start** to login.

If you are starting the Web Listener for the first time, the following window appears:

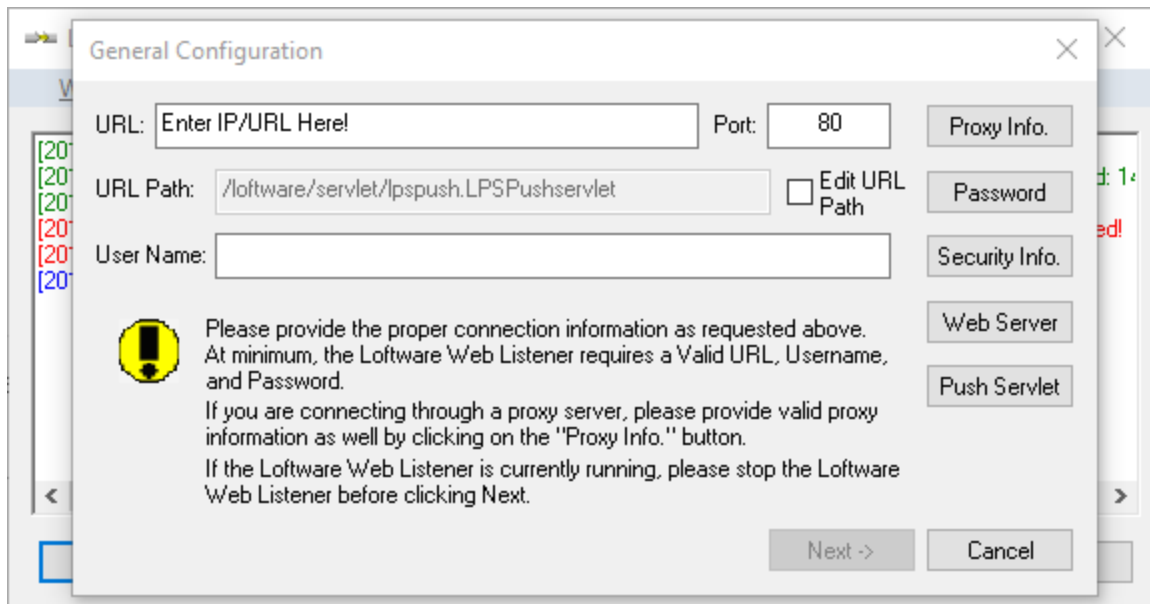


Figure 8.22: General Configuration Window

3. Enter the URL to the Software Print Server Web server, such as `http://152.22.0.58/` or `http://www.example.com/` (Port default=80). If your servlet engine is configured to use a port other than 80, enter that number.
4. Enter a valid User Name from the LPS Users list.
5. Click **Password** and type the LPS User's Password, then re-type it to confirm. Click **OK**.
6. Click **Proxy Info** ONLY if you are connecting through a Proxy Server.  
If you are NOT connecting through a Proxy Server, go to the next step.

**What is a Proxy?** A Proxy is a way to connect to the Internet in a secure fashion. A Proxy is similar to a firewall, and in Software's case, all clients going through the proxy share this extra security measure. It is generally placed between a client application and the Internet. Proxy servers are able to improve performance for groups of users, as it saves (caches) the results of all requests for a certain amount of time, therefore recalling the information is a much faster operation. Proxy servers can also be used to filter requests, for example, a proxy server could be used to prevent employees from accessing specific Web sites.

The following window appears:

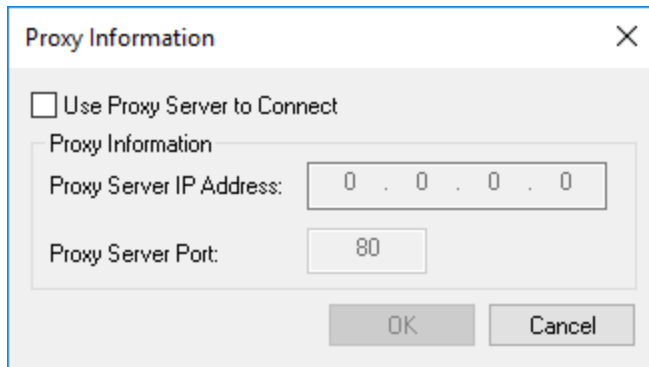


Figure 8.23: Proxy Information

- a. Select **Use Proxy Server to Connect**.
  - b. Enter your **Proxy Server IP Address** and **Proxy Server Port**.
  - c. Click **OK**. You are returned to the General Configuration dialog box.
7. Click **Security Info**. The Web Security Configuration Utility appears.

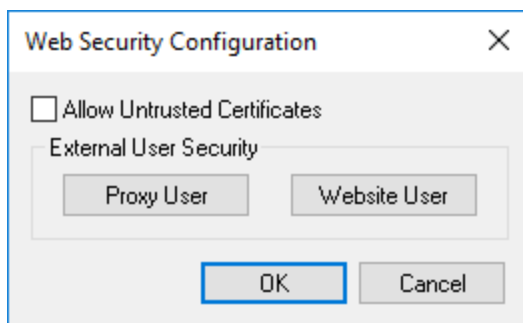


Figure 8.24: Web Security Configuration

## About the Web Security Configuration Utility

### Allow Untrusted Certificates

This gives the option of accepting certificates that are not on the system's "trusted source" list. When the Web Listener is running as a service, the security alert message that informs you that the site's security certificate is not available.

1. Choose Proxy User and/or Web site User, and complete the Security Information.
  - When would you choose both? – If you are connecting through a Proxy Server AND your Web site has security access, you must configure both Proxy User and Web Server.
  - When would you choose just one? – If you are NOT connecting through a Proxy Server, but your Web site has security access, configure just the Web site User utility. If you are only connecting through a Proxy Server, and your Web site does NOT require security access, then configure only the Proxy User utility.

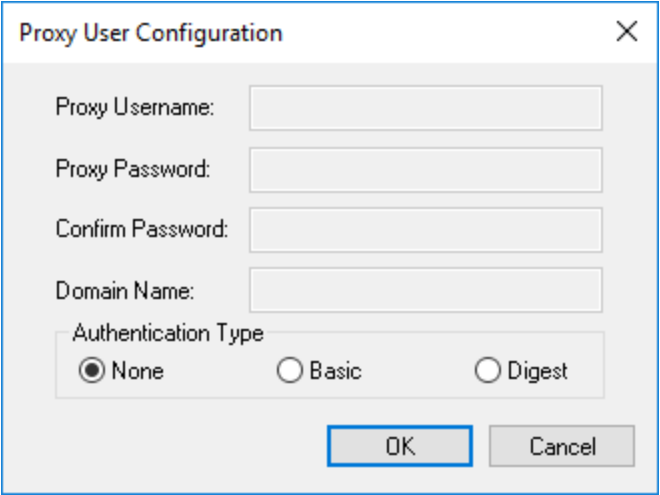
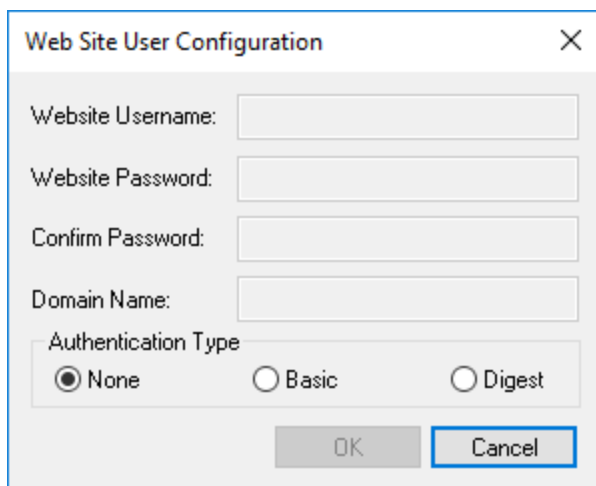
A screenshot of a 'Proxy User Configuration' dialog box. It has a title bar with a close button (X). The dialog contains four text input fields: 'Proxy Username:', 'Proxy Password:', 'Confirm Password:', and 'Domain Name:'. Below these fields is a section for 'Authentication Type' with three radio buttons: 'None' (which is selected), 'Basic', and 'Digest'. At the bottom right are 'OK' and 'Cancel' buttons. The 'OK' button is highlighted with a blue border.

Figure 8.25: Proxy User Configuration

Sections of the Proxy User Configuration Utility

Section	Description
Proxy Username	The name of the Proxy User connecting to the Web Listener.
Proxy Password and Confirm Password	The password of the Proxy User connecting to the Web Listener. Confirm the Proxy User Password by re-typing it.
Domain Name	The name of the domain the Web Listener is connecting to.
Authentication Types	<p><b>None</b> – This is the default authentication type and should be used if your Proxy Server does NOT require authentication (password, etc.) to access the Internet.</p> <p><b>Basic</b> – This is Basic (64-bit) Authentication and should be used if your Proxy Server does require the use of Basic Authentication to access the Internet.</p> <p><b>Digest</b> – This is Digest (128-bit, MD-5) Authentication and should be used if your Proxy Server requires the use of Digest Authentication to access the Internet.</p>



The image shows a 'Web Site User Configuration' dialog box with a close button (X) in the top right corner. It contains four text input fields: 'Website Username:', 'Website Password:', 'Confirm Password:', and 'Domain Name:'. Below these fields is a section for 'Authentication Type' with three radio buttons: 'None' (which is selected), 'Basic', and 'Digest'. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

Figure 8.26: Web Site User Configuration

### Sections of the Web Site User Configuration Utility

Section	Description
Website Username	The name of the Web site User connecting to the Web Listener.
Website Password and Confirm Password	The password of the Web site User connecting to the Web Listener. Confirm the Web site User Password by re-typing.
Domain Name	The name of the domain the Web Listener is connecting to.
Authentication Type	<p><b>None</b> – This is the default authentication type and should be used if your Web site does not require authentication (password, etc.) to access the Internet.</p> <p><b>Basic</b> – This is Basic (64-bit) Authentication and should be used if your Web site does require the use of Basic Authentication to access the Internet.</p> <p><b>Digest</b> – This is Digest (128-bit, MD-5) Authentication and should be used if your Web site requires the use of Digest Authentication to access the Internet.</p>

- When the Server Configuration information is complete, click **Next**.

The following is displayed:

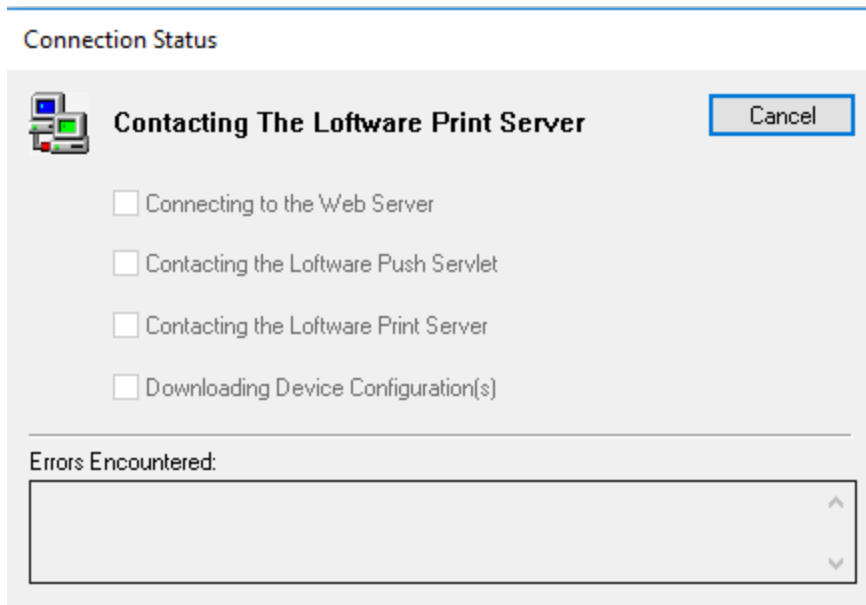


Figure 8.27: Web Listener Status Screen

If the connections are completed without error, the Web Listener Configuration Utility is displayed. If the connection fails, view the error message displayed in the Errors Encountered: section, and correct the errors. A common error might be failure to connect through the Proxy Server or typing in the wrong IP Address, either of which produces an error message.

Another common error is not having any printers configured on the Software Server. Information on adding printers to the Software Print Server is found earlier in this section.

**Note:** If the Web Listener is not available when a request to print a label is made, the XML file for the job is stored in the \spooled folder. When the Web Listener becomes available, the job will then be printed.

## Testing the Web Server and Push Servlet

**Web Server** – This button provides you with a quick way to see if the Web Server is running by opening the Web Server in your default browser. If this cannot be opened, then the Web Server is not connected.

**Push Servlet** – This button is also a quick way to see if the Web Servlet is operational. If it is, the page is displayed in your default web browser. If this page does not open, the servlet is not turned on, and this page does not display. If the servlet is turned on, but it is not connected to the Software Print Server, the page displays with a “not connected” message.

### Related Information

Information about Authentication Types may be found on the Internet in numerous locations, such as IETF. For more information, see IETF under "[External Links](#)" on page 283.



## Client-side Printer Configurations

Once the "Client-side Connections" on page 171 are established, you are required to configure your printers locally. The Web Listener does not work if the local printers are incorrectly configured. Each section of the Web Listener Configuration Utility is described below, and are important to configure and complete the Web Listener connection.

- Open Web Listener-Configuration from the **Start** menu under Software Labeling.

**Software Web Listener Configuration Utility**

**General Configuration**

URL: Valid URL Needed!

Port: 80

Username: Valid Username Needed!

Configure Connection Diagnostics Language...

**Device Configuration**

Device Name	Device Alias	Port	LLM-Number	Configuration

Configure Disable

**Log Configuration**

☐ Enable Interactive Log ☐ Enable Event Message Log

**Web Listener Logging**

☒ Enable Log Level: DEBUG

☒ Log to File \\WL\_LOG\_%Y%m%d\_%H.log

**HouseKeeping**

Log File Retention: 7 Days

OK Cancel Apply

Figure 8.28: Web Listener Configuration Utility

## General Configuration Section

Element	Description
URL	The Internet address of your Web Server that is running the Push Servlet may be written as www.loftware.com, or numerically, such as 133.23.54.78.
Port	The Port address of your Web Server. The default for most servers running http is Port 80.
UserName	The Name of the logged-in Client computer as defined in the User Database on the server.
Proxy Server	URL for Proxy Server as described previously. (If using a Proxy Server)
Configure Connection	When clicked, allows you to re-configure the connection.
Diagnostics	Click to open a <a href="#">Diagnostics window</a> for troubleshooting if you are experiencing problems with the Web Listener.

## Device Configuration Section

This list is generated from the User Database on the LPS. Only the devices for the connected User are displayed.

Element	Description
Device Name	The name of the Loftware device to which you are connected.
Device Alias	The Alias assigned to this device.
Port	The Port denotes how the device is connected (TCP/IP or Local [i.e.; COM1] Connection). This column is not populated until the port has been configured on the Listener side.
LLM-Number	The number of the device as listed on the Server-side.
Configuration Status	True indicates that the device is configured on the Listener-side; False indicates that this device has not been configured on the Listener-side.
Configure	Opens the Device Configuration dialog box to configure a device. This is grayed out until a device is selected.
Disable	Disables the selected device from the Client-side device list. This is a helpful choice when a device has been deleted from the Server-side device list. The Disable button, however, does not affect the Server Side list. This is grayed out until a device is selected.

## Web Listener Logging Configuration Section

**Enable Event Message Log** – When selected, specific and detailed event information is sent to the Event Log. This log can be viewed with the Event Viewer, accessed from: Start | Settings | Control Panel | Administrative Tools | Event Viewer

Element	Description
Enable	When selected, a detailed and specific log file is written to the Web Listener window. If it is left cleared, only basic messages are displayed such as start, stop, and basic error messages. This only applies to Interactive Mode.

Element	Description
Log Level	<p>Sets the type of log messages to include.</p> <p><b>DEBUG</b> - Includes everything in INFO, WARNING, and ERROR plus the following:</p> <ul style="list-style-type: none"> <li>■ Verbose Security</li> <li>■ Display Files</li> <li>■ Client Debug</li> <li>■ Client Connections</li> <li>■ Client Audit</li> <li>■ Client SQL</li> </ul> <p><b>INFO</b> - Includes the following information.</p> <ul style="list-style-type: none"> <li>■ Log Jobs</li> <li>■ Checkpoint</li> <li>■ Settings/Configurations</li> <li>■ Job Numbers</li> <li>■ Warnings and Errors</li> </ul> <p><b>WARNING</b> - Includes ERRORS.</p> <p><b>ERRORS</b> -Limited to only application errors.</p>
Log to File	<p>Logs the information displayed in the Software Print Server Interactive window to a file. Refer to the <a href="#">Formatting and Locating Log Files</a> section in this topic for more information.</p>

## Diagnostics Window Dialog

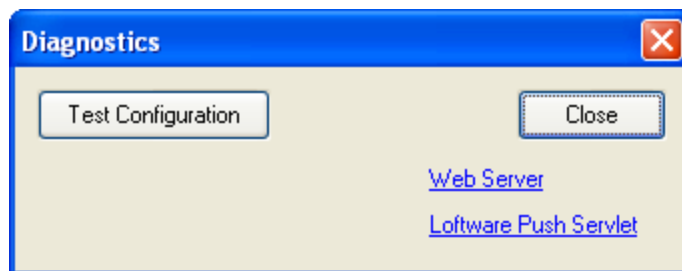


Figure 8.29: Diagnostics Window

## Diagnostics Window Elements

Element	Description
Test Configuration	This starts the Connection Status process.
Web Server	This hypertext link provides you with a quick way to see if the Web Server is running by opening the Web Server in your default browser. If this cannot be opened, then the Web Server is not connected.

Element	Description
Software Push Servlet	This hypertext link is also a quick way to see if the Web Servlet is operational. If it is, the page is displayed in your default web browser. If this page does not open, the servlet is not turned on, and this page does not display. If the servlet is turned on, but it is not connected to the Software Print Server, the page displays with a "not connected" message.

## Formatting and Locating Log Files

You can set the format for your On Screen and Performance Data log files using formatting codes described in the following tables.

### For Example

The **Log to File** setting, `\\WD_LOG_%Y%m%d_%H.log`, creates a file called "WD\_LOG\_20100922\_11.log" in the Logs | 2010 | 09 | 22 folder at 11:30 AM on September 22, 2010.

The following table contains data from the Microsoft MSDN Library.

Formatting Code	Formatting Description
%a	Abbreviated weekday name
%A	Full weekday name
%b	Abbreviated month name
%B	Full month name
%c	Date and time representation appropriate for locale
%d	Day of month as decimal number (01 – 31)
%H	Hour in 24-hour format (00 – 23)
%I	Hour in 12-hour format (01 – 12)
%j	Day of year as decimal number (001 – 366)
%m	Month as decimal number (01 – 12)
%M	Minute as decimal number (00 – 59)
%p	Current locale's A.M./P.M. indicator for 12-hour clock
%S	Second as decimal number (00 – 59)
%U	Week of year as decimal number, with Sunday as first day of week (00 – 53)
%w	Weekday as decimal number (0 – 6; Sunday is 0)
%W	Week of year as decimal number, with Monday as first day of week (00 – 53)
%x	Date representation for current locale
%X	Time representation for current locale
%y	Year without century, as decimal number (00 – 99)

Formatting Code	Formatting Description
%Y	Year with century, as decimal number
%z, %Z	Either the time-zone name or time zone abbreviation, depending on registry settings; no characters if time zone is unknown
%%	Percent sign

## Configuring Devices on the Client-side

1. Click a device in the **Device Configuration** section, and click **Configure**. The **Device Configuration** dialog box appears.

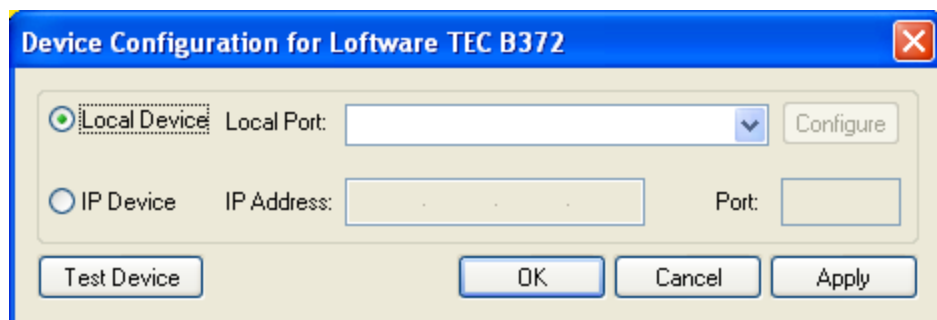


Figure 8.30: Listener Client Printer Configuration

2. Configure a Port for the printer. Select a **Local Port**, such as COM 1, LPT 2, USB from the drop-down list, or choose the appropriate IP Address, and click **Apply**. The **Test Device** button becomes active.

**Note:** USB Ports are ONLY listed in the Local Port list if the USB Printer is connected and is powered on.

3. Click **Test Device**, and a short print stream is sent to the configured device.

If successful, the device prints the print stream, if not, then check the device to make sure the device is powered on, that the Port is working, etc.

## More Device Configuration Information

In Interactive Mode, you can check to see what Local Ports are available for the Web Listener by choosing Options | Check Local Ports from the menu bar. A box is displayed that lists the available local ports (i.e., COM 1, LPT 2, USB, etc.) You may use one of these ports or a TCP/IP Port as shown in the Configure section. This is a diagnostics tool and is not a necessary step.

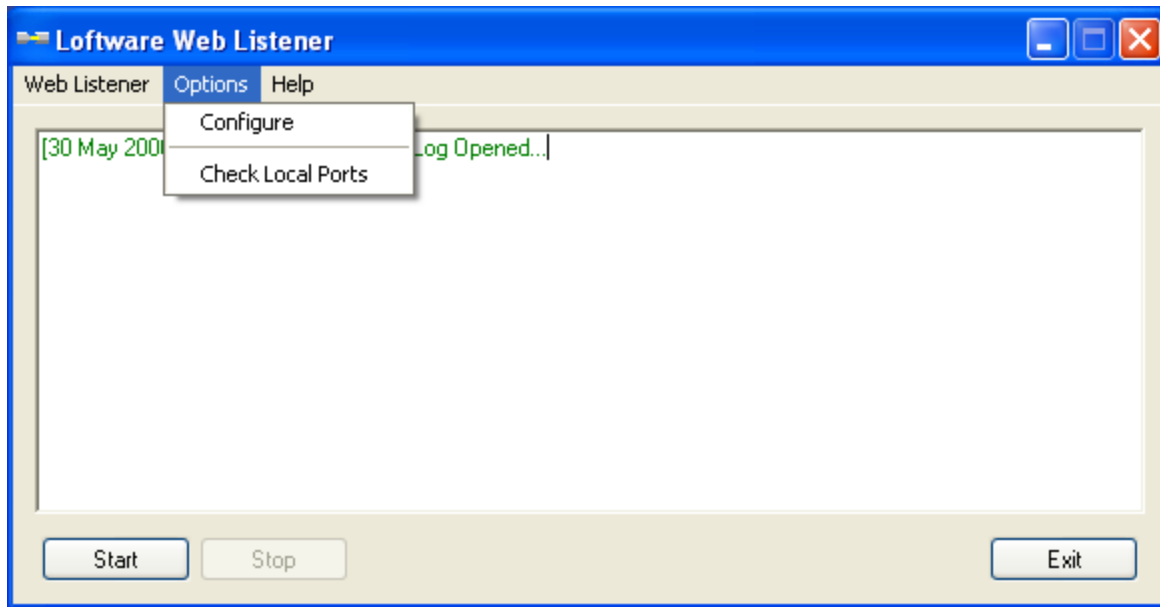


Figure 8.31: Web Listener, Interactive Mode.

You can also access the Web Listener Configuration Utility in Interactive Mode by clicking Options | Configure on the menu bar.

#### Related Information

See the Device Connections section of the *Software Label Manager User's Guide* for more USB information.

See the *Device Connections* and *Printers and Labels* sections of the *Software Label Manager User's Guide* for more documentation on printer connections and error messages.

## Using Web Listener as a Service

### Starting Web Listener as a Service

Select Start | Settings | Control Panel | Administrative Tools | Services | Software Web Listener

**Note:** Initially, Web Listener should be started interactively to configure devices; thereafter, it should be run as a service.

### Using Web Listener as a Service

A service is an application that runs in the background, unseen, and processes information without user login. See the LPS section of this guide for more information on the LPS as a service. Web Listener runs as a service, staying connected to the Web Server and the LPS with the help of the LPSPushservlet, as described previously. A file dropped to the LPS is processed by the Listener Client and printed if the configurations have been followed correctly.

**Note:** If the Web Listener is not available when a request to print a label is made, the XML file for the job is stored in the \spooled folder. When the Web Listener becomes available, the job will then be printed.

The Web Listener is designed from the ground up to run as a service with all applicable security. It can be configured to run when the computer is booted. It runs at a lower level (ring) than ordinary programs and cannot be seen or changed by the operator.

Services perform their functions without requiring the operator to log on, thus providing protection from intentional or accidental change (security).

---

## The ActiveX Client Control

Programmers developing in 32-bit languages supporting ActiveX Controls can easily interface Loftware's barcodes printing modules directly into their own applications. Loftware's Client Control reduces the level of knowledge and expertise required to connect and print to stand-alone and networked barcodes label printers. 32-bit development languages, such as Visual Basic, Delphi, Access, Power Builder, Visual C++, etc., can utilize this technology.

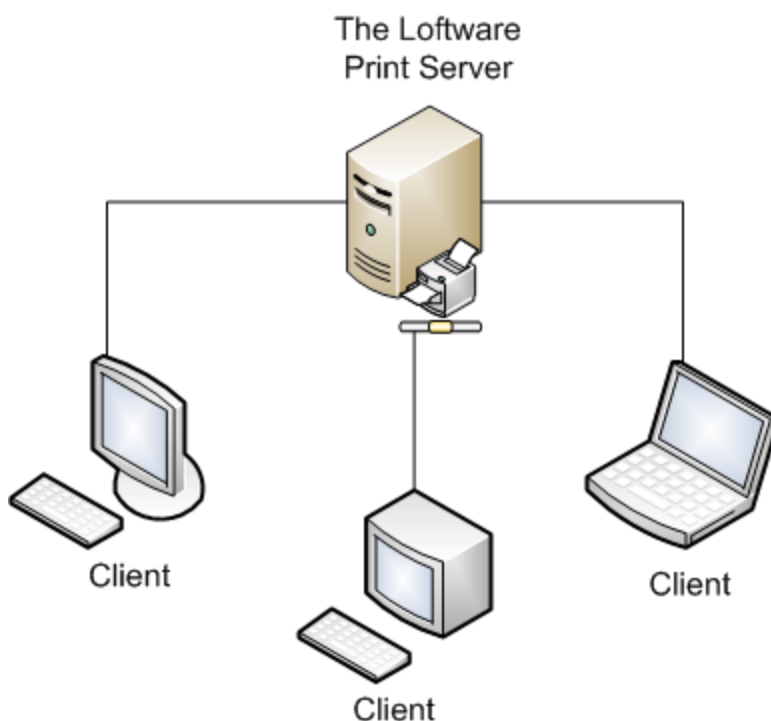


Figure 9.1: Computers using one of the Client Controls to offload print jobs to the LPS

This section documents the Loftware ActiveX Client Control. This control has a thin footprint because it does not require that the Loftware Label Manager subsystem be installed on the same computer as the control. It is called the Client Control because it acts as a client to the LPS (Loftware Print Server). Use this control when your application is running in several places, needs to access many printers, and requires a small footprint.

As long as you do not initialize the control by using any of the methods, or the `isRunning` Property, it does not look for the Loftware Print Server. This allows you to embed the control in your program and only use it when it is turned on either by the user or programmatically.



The ActiveX Client Control has the ability to connect to the LPS through a socket connection, similar to our On-Demand Print Client and our Status Client. If you are using an older version of the ActiveX Client Control with a custom application, you can upgrade to the latest version of ActiveX, which is backwards compatible. In addition, your custom application can be converted to use the new socket connection with minimal modifications. To set up your application using the ActiveX Client Control to connect to the LPS through sockets, review information found under Login Method and under ClientIniPath Property.

## Installation and Use of ActiveX Client Control

1. Install the Software Print Server (LPS).

**Note:** The LPS can be installed on any computer with a supported operating system anywhere on your LAN or WAN.

2. Perform a client install of the ActiveX Control.
3. Configure all connected printers on the Server.
4. Design the necessary label formats on the server.

**Note:** You may also design labels on any client computer and either share the files on the server or copy the files to the server when done. The Options | File Locations menu in Software Label Manager allows you to set network locations for your design files. This allows you to design on a client computer while your labels are saved to the computer specified, usually the LPS computer.

5. Verify that the LPS is working by building a simple label and manually creating a .pas file for it. Copy your pas file to the directory that LPS is scanning and your label should print.

### Related Information

For information on configuring printers, refer to *Device Connections* in the *Software Label Manager User's Guide*. For information on installing, configuring, and troubleshooting the LPS, refer to *The Software Print Server* in this guide.

## Design Scenario and Distribution

Suppose that you have written an application that you sell to end-user accounts. These users would typically install your application on several client computers. Your application may also be a server-based application.

You want to add the ability to print barcodes labels from your application, but you do not want to have to pay for it unless your customer needs the barcoding feature. You need a solution that you can embed in your program that only has to be paid for if used.

You can do just that with the ActiveX Client Control. ActiveX Client Control is designed in such a way that you can embed it into your program and distribute it to your customers as part of your application. When your customer decides that they want to use the barcoding capability of your application, you:

1. Have the customer upgrade to the Software product that contains the Software Print Server.
2. Install the Software Print Server on the customer network.
3. Activate the control in your program using a setup screen, registry setting, or .ini file.

**Note:** You may install the Software Print Server on the same computer as your application. However, this may slow system performance if both applications are busy.

**Note:** The ActiveX Client Control has a thin footprint and does not require the Software Print Server to be installed on the same computer. The client control does require that the LPS be installed somewhere on the network.

## Run the Client X All Methods.exe Sample

By running the sample program, called Client X All Methods.exe, and examining the code, you will be able to see how each property, method and event of the ActiveX control is used. Sample programs are installed to Software Labeling\Sample Programs with the Software Print Server Premier Edition.

Follow the instructions below for the ability to run the Client X All Methods.exe program on a computer that has the Software Print Server (LPS) installed or on a client computer.

1. Install the Software Print Server.
2. Perform a client install of the Active X control as described in the ActiveX Section of this guide.
3. Rename the llmwcInt.sav file on the LPS to llmwcInt.ini. Verify the information in the file is accurate.
4. To run the program from a Client computer, install the LPS Clients from the LPS Client Install folder on the LPS.
5. Place Client X All Methods.exe on the client computer.

**Important!** Before running this program on the Client computer, verify the account logged onto the Client computer has permission to access the Software Labeling directory on the Software Print Server computer with Full Control. On an NTFS partition, the permissions must be set at the file level. If the permissions are not set correctly, the error message Write Access Denied may be displayed and the program will exit.

6. Double-click the Client X All Methods.exe.
7. Follow the directions on the form to print labels. A pas file will be generated and dropped in the designated Scan directory.

### Related Information

For more explanation of the properties, methods and events, refer to the *ActiveX* section of the *Software Print Server User's Guide*.

For complete installation instructions and system requirements, refer to the *Installing* section of the *Software Label Manager User's Guide* and *The Software Print Server* section of the *Software Print Server User's Guide*.

## Using ActiveX Client Control

1. Install the ActiveX Client Control on a client computer.
2. Load a sample program, learn, and study it.

**Note:** Sample programs are installed with the LPS. They are also present in a subdirectory called sample programs.

3. Create a project in a programming tool that can utilize ActiveX Controls.
4. Add the Software ActiveX Client Control to the new project
5. Initialize the control by setting the PrinterPath, DropDirectory, .LabelsPath, and LayoutPath properties.
6. Choose a label by invoking the SetLabelName Method.
7. Set field data by using the SetData Method.
8. Print the label format directly by using the PrintJob Method.

**Note:** If your label does not print, try stopping the LPS scanning process to see if the control writes a .pas file to the directory specified with the DropDirectory Property.

## Troubleshooting the ActiveX Client Control

**Important:** It is critical that you trap error events. This is especially true if the Software Print Server is running in a clustered environment and a failover occurs. Many of the ClientX methods throw critical errors during the failover transition.

- Do not set any Path properties of the control until you have a valid LPS installation. The control throws ErrorEvents if it cannot find the paths that you set.
- Make sure that you have some labels designed and printers configured on the LPS before trying to use the PrintJob Method.
- Make sure you trap errors and display the error string in the ErrorEvent. This saves considerable debug time.
- Verify that the LPS is working by building a simple label and manually creating a PAS file for it. Copy your PAS file to the directory that the LPS is scanning and your label should print.
- You may want to add a multi-line list box to your application that you can add a line to for each WarningEvent and InfoEvent. This provides valuable information that helps you get up and running quickly. You can remove these events when they are no longer needed.
- ActiveX Client control does not allow you to assign a drop directory to any directory that does not contain the subdir /status. The result is an error dialog. If you try to execute the Visual Basic application anyway, it shuts down, and you must end the task from the Visual Basic program, reload it, and revert to the last code save.

### Related Information

To troubleshoot PAS file printing, refer to *The Software Print Server* section in this guide.  
For more information on ClientX errors, refer to the *ActiveX Error Event* section in this guide.

## Distributing the ActiveX Client Control with your Application

The Client Control can be distributed to anyone wishing to use the features that it provides. When building an install program for your application, include the required Software files with it. This way, your setup program takes care of installing the Software control and its associated files, thereby avoiding having to run two setup programs.

The following files are needed to use the ActiveX Client Control. If your programming language has a setup wizard, it probably picks up the correct files by scanning our dependency file.

- ATLClient.tlb -> winsys folder
- ClientX.dll(the control) -> winsys folder. This file needs to be registered.

## ActiveX Client Control Properties

### ClientIniPath Property

#### Syntax

```
Object.ClientIniPath = path
Type = string, read/write
```

#### Description

The ClientIniPath property allows the use of an "LLMWCInt.ini" file for the control to read the properties from the LPS. Upon setting this property, the file is parsed and an internal mapping of all servers and corresponding printers takes place. The llmwclnt.ini file needs to be created in a directory somewhere on your network. The "LabelsPath," "LayoutPath," "PrinterPath," "DropDirectory" as well as "ServerName" and "ServerAlias" properties for multiple servers are specified in the .ini file. If you are only using one Software Server (LPS), you need not use this property, and the other properties listed above can be set in code. The following syntax example shows an .ini file listing two LPS servers. Setting the "ClientIniPath" property to point to this file causes the control to be set up for multiple LPS servers. The ServerCount property is actually equal to 3 after setting the ClientIniPath to this .ini file. Server 0 is the default server that existed before the setting of ClientIniPath. Server 1 and 2 correspond to the ones listed in the .ini file.

The LLMWCInt.ini file can also be used to connect to multiple servers through socket connections. When used in this way, it is not necessary to specify the "Labels Path," "Layout Path," "Printer Path," "Drop Directory" or "ServerAlias" properties in the .ini file, only the "Server Name" and the "IPAddress" property are required. Any application using the ActiveX Control automatically logs in to all the servers listed in the LLMWCInt.ini upon execution of the program. Once you are connected to multiple servers, you can use the "SetServer" Method to move between servers.

**Note:** This property is what allows the Client Control to be aware of multiple Software Print Server Installations. See Performance Considerations in the Installation Guide for information to help you to decide when you need to divide your printing load among multiple servers.

#### Syntax of llmwcInt.ini file:

```
[Receiving1]
Name=JANAA
Alias=Jana Computer
LabelsPath=c:\program files\Software Labeling\labels
LayoutPath=c:\program files\Software Labeling\Layouts
PrinterPath=c:\program files\Software Labeling
ScanPath=c:\program files\Software Labeling\wddrop
[DemoRoom]
Name=TRAINING
Alias=Training Room Server
LabelsPath=\\training\Software Labeling\labels
LayoutPath=\\training\Software Labeling\layouts
PrinterPath=\\training\Software Labeling
ScanPath=\\training\Software Labeling\wddrop
```

#### Example

```
'get a list of LPS servers and populate a list box with their names and
'aliases. This List could later be used to decide which server to set with the
'SetServer Method.
```

```
On Error GoTo Handler
frmFront.ClientX1.ClientIniPath = Trim(txtClientIniLocation.Text)
For i = 1 To frmFront.ClientX1.ServerCount - 1
    frmFront.ClientX1.SetServer (i)
    lstServers.AddItem frmFront.ClientX1.ServerName & " ALIAS " &
frmFront.ClientX1.ServerAlias
Next i
```

#### Syntax of llmwcInt.ini file using Socket Connections

```
[Shipping1]
Name=SHIPLINE1
Address=165.10.0.120
[Receiving1]
Name=RECEIVING1
Address=165.10.0.122
```

## DropDirectory Property

**Important:** This property has been deprecated. It is included in the Software Print Server for backward compatibility.

### Syntax

```
dropDir = Object.DropDirectory
Type = string, read/write if not using ClientIniPath, otherwise, read only
```

## Description

The DropDirectory reflects the path to the Server's LPS scan path. This path is specified with a mapped drive or a UNC. Files are deposited in the scan directory for the Loftware Print Server to process when the PrintJob method is invoked. A good way to verify if your program is working properly is to shut down the LPS on the server and use explorer to view the folder specified with the DropDirectory property. A file is displayed after requesting a label with the PrintJob method with a .pas extension. View this file with an ASCII editor to verify that it is syntactically correct.

**Note:** If the LPS has multiple scan directories, it is up to your program to choose which directory to use for a given print job.

### Example

```
' Set the DropDirectory for the Default server (0)
ClientX1.DropDirectory = "\\XFILES\LOFTWARE\WDDrop"
```

## Duplicates Property

### Syntax

```
Object.Duplicates = short
Type = short, read/write
```

### Description

The Duplicates property is a read/write property that sets the amount of duplicate labels to print. The default for this property is one. Duplicate labels are EXACT copies of the original label. When using this property with a label that has an incrementing or decrementing serial number, this many labels print before incrementing (or decrementing) the number. To create multiple labels with unique serial numbers use the Quantity Property.

### Example

```
'print 2 copies of each serial number 5 times.
'the total number of labels printed = 10
ClientX1.SetLabelName "Label1.lwl"
ClientX1.SetData 0, "ABC-123"
ClientX1.Quantity = 5
ClientX1.Duplicates = 2
ClientX1.PrintJob
```

## FieldCount Property

### Syntax

```
Object.FieldCount
Type = short, read/write
```

## Description

The FieldCount, FieldName, and FieldLength properties describe the array created when the SetLabelName method is invoked. It is important to understand that this array is the key to having access to all fields in your label in a dynamic fashion. Fields in the array are accessed by their index number in the array or by the field name itself. Your program may not know ahead of time which and how many fields are in the label. This is why you need to iterate through the array with an index. If you do know your field names ahead of time, it is easier to set their data using the FieldName property.

The FieldCount property is a read only property that displays how many variable fields there are in the current label format. This allows you to iterate through the field array that is generated when the SetLabelName method is invoked. There is no default for this property.

**Example**

```

'This example builds an SQL statement on the fly based on the field names
'in the label. The database is then hit with a random key field and the
'data for the fields is set. See the "Interface" section of Section 1 for
'more information on providing data for the general case.
'Assumptions
'The field names in the label format are constrained to the field names in
'the database. (See Section 1.)
'Global myDatabase As Database defined in module1
'Global myRecordset As Recordset defined in module1
'assume softwarePath has been preset to the network location of the
'Software print server (LPS).
'GScanDirectory has been preset to the directory where the LPS is scanning.
Public Sub populateControlWithRandomData()
Dim sqlStatement As String, i As Integer, thisFieldName As String
'initialize the control to point to the LPS
ClientX1.LabelsPath = softwarePath & "\\labels"
ClientX1.LayoutPath = softwarePath & "\\layouts"
ClientX1.DropDirectory = gScanDirectory
ClientX1.PrinterPath = softwarePath
'open the database
Set myDatabase = OpenDatabase(App.Path & "\\sample.mdb")
'build SQL statement to grab data for this label
sqlStatement = "SELECT "
ClientX1.SetLabelName "mytest.LWL"
For i = 0 To ClientX1.FieldCount - 1
    sqlStatement = sqlStatement & "[" & ClientX1.FieldName(i) & "]" & ", "
Next i
'get rid of the last comma before the FROM clause and append key
sqlStatement = Left(sqlStatement, Len(sqlStatement) - 2) & " FROM Newwar WHERE
NAME1='" & cboRecordChoice.Text & "';"
'grab the record and populate the data
'only grab the fields we need for this label from the database
Err = 0
On Error Resume Next
Set myRecordset = myDatabase.OpenRecordset(sqlStatement)
If Err <> 0 Then
MsgBox "SQL Error #" & Err & " SQL = " & sqlStatement, vbInformation, "SQL Error"
Exit Sub
End If
' populate the label fields with the retrieved data
For i = 0 To ClientX1.FieldCount - 1
    thisFieldName = ClientX1.FieldName(i)
    ClientX1.SetData thisFieldName, myRecordset.Fields(thisFieldName)
Next i
myRecordset.Close
pickRandomRecord
myDatabase.Close
'print the label
ClientX1.PrintJob
End Sub

```



## FieldLength Property

### Syntax

```
Object.FieldLength (index) or (FieldName)
Type = short, read
```

### Description

The 'FieldCount', 'FieldName', 'Field Data', and 'FieldLength' properties describe the array created when the 'SetLabelName' method is invoked. It is important to understand that this array is the key to having access to all fields in your label in a dynamic fashion. Fields in the array can be accessed by their index number in the array or by the field name itself. Your program may not know ahead of time which and how many fields are in the label. This is why we allow you to iterate through the array with an index. If you do know your field names ahead of time, it is easier to set their data using the 'FieldName' property.

The FieldLength property is a read only property that displays the length of a specified field in the current label format. This property can be retrieved by the actual field name or field index number. This property is not filled until the SetLabelName method has been invoked.

**Note:** This property is very useful for pre-verifying data before actual printing by only allowing the amount of characters in the actual label field to be entered.

There is no default for this property.

#### Example

```
'By Field Index Number:
Text1.MaxLength = ClientX1.FieldLength 0
'By Field Name:
Text1.MaxLength = ClientX1.FieldLength "PARTNUMBER"
```

## FieldName Property

### Syntax

```
Object.FieldName ( index )
Type = short, read/write
```

### Description

The 'FieldCount', 'FieldName', 'Field Data', and 'FieldLength' properties describe the array that is created when the 'SetLabelName' method is invoked. It is important to understand that this array is the key to having access to all fields in your label in a dynamic fashion. Fields in the array can be accessed by their index number in the array or by the field name itself. Your program may not know ahead of time which and how many fields are in the label. This is why we allow you to iterate through the array with an index. If you do know your field names ahead of time, it is easier to set their data using the 'FieldName' property.

The `FieldName` property is a read only property that displays the name of a specific field in the current label format. This property can only be retrieved by the field index number. This property is not filled until the `SetLabelName` method has been invoked. There is no default for this property.

**Example**

```

'This example builds an SQL statement on the fly based on the field names
'in the label. The database is then hit with a random key field and the
'data for the fields is set. See the "Interface" section of Section 1 for
'more information on providing data for the general case.
'Assumptions
'The field names in the label format are constrained to the field names in
'the database. (See Section 1)
'Global myDatabase As Database defined in module1.
'Global myRecordset As Recordset defined in module1
'assume softwarePath has been preset to the network location of the
'Loftware print server (LPS).
'GScanDirectory has been preset to the directory where the LPS is
'scanning.
Public Sub populateControlwithRandomData()
Dim sqlStatement As String, i As Integer, thisFieldName As String
'initialize the control to point to the LPS
ClientX1.LabelsPath = softwarePath & "\\labels"
ClientX1.LayoutPath = softwarePath & "\\layouts"
ClientX1.DropDirectory = gScanDirectory
ClientX1.PrinterPath = softwarePath
'open the database
Set myDatabase = OpenDatabase(App.Path & "\\sample.mdb")
'build SQL statement to grab data for this label
sqlStatement = "SELECT "
ClientX1.SetLabelName "mytest.LWL"
For i = 0 To ClientX1.FieldCount - 1
    sqlStatement = sqlStatement & "[" & ClientX1.FieldName(i) & "]" & ", "
Next i
'get rid of the last comma before the FROM clause and append key
sqlStatement = Left(sqlStatement, Len(sqlStatement) - 2) & " FROM Newwar WHERE
NAME1='" & cboRecordChoice.Text & "';"
'grab the record and populate the data
'only grab the fields we need for this label from the database
Err = 0
On Error Resume Next
Set myRecordset = myDatabase.OpenRecordset(sqlStatement)
If Err <> 0 Then
    MsgBox "SQL Error #" & Err & " SQL = " & sqlStatement, vbInformation, "SQL Error"
    Exit Sub
End If

'populate the label fields with the retrieved data
For i = 0 To ClientX1.FieldCount - 1
    thisFieldName = ClientX1.FieldName(i)
    ClientX1.SetData thisFieldName, myRecordset.Fields(thisFieldName)
Next i
myRecordset.Close
pickRandomRecord
myDatabase.Close
'print the label
ClientX1.PrintJob
End Sub

```

## isRunning Property

### Syntax

```
Object.isRunning = boolean
Type = boolean
```

### Description

The isRunning property checks to see if the currently selected Software Print Server is scanning. If the LPS is not found, or is not scanning, error # 25518 is thrown in the ErrorEvent. Make sure that you handle this error in the ErrorEvent. If you do NOT, your program may crash.

#### Example

```
On Error Resume Next
Err = 0
If ClientX1.isRunning = False Then
    MsgBox("LPS is not running!")
End If
If Err Then
    If (Err.Number - &H80000000 = 25518) then
        MsgBox "LPS is not reachable"
    End If
End If
```

## JobName Property

### Syntax

```
jobName = Object.JobName
Type = string, read/write
```

### Description

JobName is a read/write property reflecting a unique identifier for the current job. The default value changes after the PrintJob method is called and follows this naming convention:

ComputerName + "X" + unique instance number + "\_" + YYYYMMDDHHNNSS + serial number per second

If you wish to specify your own JobName, it is up to the creator to ensure its uniqueness across time as well as space. It is through this identifier that future job status is returned. The JobName is also reflected in the Status View console that monitors the status and progress of the jobs processed by the LPS. This property is for feedback purposes only. It is not needed to print labels.

#### Example

```
Dim jobName as string
' Get the current JobName to store for reference
jobName = ClientX1.JobName
```

## LabelsPath Property

**Important:** This property has been deprecated. It is included for backward compatibility.

### Syntax

```
LabelsPath = Object.LabelsPath
Type = string, read/write if not using ClientIniPath, otherwise, read only
```

### Description

LabelsPath is a read only property for servers listed in the LLMWCInt.ini file and writeable for the default server (Server 0). The LabelsPath reflects the UNC or mapped drive path to the labels directory that the server is using. When LabelsPath is used, the LabelsPath is prepended to label files in the call to SetLabelName if the path is not present.

If you receive “Format not found” errors, chances are the LabelsPath is incorrect or not mapped properly from the client. Error number 25513 may be thrown if the path does not exist. Handle this error in the ErrorEvent to prevent your program from crashing!

#### Example

```
' Set the LabelsPath for the Default server (0)
ClientX1.LabelsPath = "\\XFILES\LOFTWARE\Labels"
```

## LayoutPath Property

**Important:** This property has been deprecated. It is included for backward compatibility.

### Syntax

```
layoutPath = Object.LayoutPath
Type = string, read/write if not using ClientIniPath, otherwise, read only
```

### Description

The LayoutPath reflects the UNC or mapped drive path to the Server's Layout directory. LayoutPath is currently not used by the control and is reflected due to its existence in the LLMWCInt.ini file.

#### Example

```
' Set the LayoutPath for the Default server (0)
ClientX1.LayoutPath = "\\XFILES\LOFTWARE\Layouts"
```

## Pages Property

### Syntax

```
Object.Pages
Type = short, read/write
```

## Description

The Pages property is a read/write property that sets how many pages of labels are printed. A page of labels is a copy of an entire page of labels created when printing with layouts. The default for this property is one. For more detail on pages of labels, consult the LLM-WIN manual. You need not use this property if your label does not use a "multi-up" layout.

### Example

```
'print 2 identical pages, 10 labels/page with 2 labels for each serial #
ClientX1.SetLabelName "Label1.lwl"
ClientX1.SetData 0, "ABC-123"
ClientX1.Quantity = 5
ClientX1.Duplicates = 2
ClientX1.Pages = 2
ClientX1.PrintJob
```

## PrinterAlias Property

### Syntax

```
printerAlias = Object.PrinterAlias
Type = string, read only
```

### Description

PrinterAlias is used along with PrinterName, PrinterCount, and PrinterPort properties to allow you to prompt your users for the target-configured printer to whom they wish to print. Essentially, these properties expose the list of configured printers that was set up in the Loftware Label Manager. Printer Alias is a descriptive name for a printer that is assigned when in the device configuration menu of Loftware Label Manager (same dialog box where you specify a port/spool/IP address).

**Note:** The PrinterPath or the ClientIniPath property MUST be set in order to obtain the printer information. The Printer properties only reflect the printers that have been previously configured at the LPS. They are Read Only!

**Example**

'this example populates a list box with printer aliases. If the  
'printer alias does not exist, it uses the default printer name.

```
Public Sub populatePrinterList()
Dim i As Integer
If ClientX1.PrinterCount = 0 Then
    MsgBox "You have not configured any printers in Loftware yet."
    Exit Sub
End If
For i = 1 To ClientX1.PrinterCount
    If ClientX1.PrinterAlias(i) = "" Then
        frmFront.lstPrinters.AddItem ClientX1.printerName(i)
    Else
        frmFront.lstPrinters.AddItem ClientX1.PrinterAlias(i)
    End If
Next i
End Sub
```

## PrinterCount Property

### Syntax

```
numConfiguredPrinters = Object.PrinterCount
Type = short, read only
```

### Description

PrinterCount is used to return the number of printer seats on the license key. Along with PrinterName, PrinterPort, and PrinterAlias properties, it provides information on the configured printers to which users wish to print. These properties expose the list of configured printers that are set up in the Loftware Label Manager as CLIENT DEFINED.

**Note:** Either the PrinterPath or the ClientIniPath property MUST be set in order to obtain the printer information. The Printer properties only reflect the printers that have been previously configured at the LPS. They are Read Only!

**Example**

```
'this example populates a list box with printer aliases. If the printer
'alias does not exist, it uses the default printer name.
Public Sub populatePrinterList()
Dim i As Integer
If ClientX1.PrinterCount = 0 Then
    MsgBox "You have not configured any printers in Software Yet."
    Exit Sub
End If
For i = 1 To ClientX1.PrinterCount
    If ClientX1.PrinterAlias(i) = "" Then
        frmFront.lstPrinters.AddItem ClientX1.printerName(i)
    Else
        frmFront.lstPrinters.AddItem ClientX1.PrinterAlias(i)
    End If
Next i
End Sub
```

## PrinterName Property

### Syntax

```
printerName = Object.PrinterName
Type = string, read only
```

### Description

PrinterName is used along with PrinterCount, PrinterPort, and PrinterAlias properties to allow you to be able to prompt your users for the target-configured printer to which they wish to print. Essentially, these properties expose the list of configured printers that was set up in the Software Label Manager.

**Note:** The PrinterPath or the ClientIniPath property MUST be set in order to obtain the printer information. The Printer Properties only reflect the printers that have been previously configured at the LPS. They are Read Only.

**Example**

```
'this example populates a list box with printer aliases. If the printer
'alias does not exist, it uses the default printer name.
Public Sub populatePrinterList()
Dim i As Integer
If ClientX1.PrinterCount = 0 Then
    MsgBox "You have not configured any printers in Software Yet."
    Exit Sub
End If
For i = 1 To ClientX1.PrinterCount
    If ClientX1.PrinterAlias(i) = "" Then
        frmFront.lstPrinters.AddItem ClientX1.printerName(i)
    Else
        frmFront.lstPrinters.AddItem ClientX1.PrinterAlias(i)
    End If
Next i
End Sub
```



## PrinterPort Property

### Syntax

```
printerPort = Object.PrinterPort
Type = string, read only
```

### Description

PrinterPort is used along with PrinterName, PrinterCount, and PrinterAlias properties to allow you to be able to prompt your users for the target-configured printer to which they wish to print. Essentially, these properties expose the list of configured printers that is set up in the Loftware Label Manager.

**Note:** Either the PrinterPath or the ClientIniPath property MUST be set in order to obtain the printer information. The Printer Properties only reflect the printers that have been previously configured at the LPS. (They are Read Only!)

#### Example

```
'this example populates a list with configured printer ports
Public Sub populatePorts()
Dim i As Integer, gotOne As Boolean
For i = 1 To ClientX1.PrinterCount
    If ClientX1.printerName(i) <> "Not Configured" Then
        frmFront.lstPorts.AddItem ClientX1.PrinterPort(i)
    End If
Next i
End Sub
```

## PrinterNumber Property

### Syntax

```
Object.PrinterNumber = short
Type = string, read/write
```

### Description

The PrinterNumber property is a read/write property that sets which label formats to print to which configured LPS printer. The default for this property is configured Printer #1. Using the PrinterName or PrinterAlias properties, you could prompt your user with a better description of the target printer. You could then resolve their choice into a number, which ultimately must be used in code to specify the printer.

**Example**

```
'Printer 1 in Software might be defined as: Software Intermec 3440 on COM2:
'Printer 2 in Software might be defined as: HP LaserJet 4M on \\Server\Print\HP
'In this example, to print to the Intermec printer, set the PrinterNumber
'property to 1.
ClientX1.SetLabelName "Label1.lwl"
ClientX1.SetData 0, "ABC-123"
ClientX1.PrinterNumber = 1
ClientX1.PrintJob
'To print to the HP printer, set the PrinterNumber property to 2:
ClientX1.SetLabelName "Label2.lwl"
ClientX1.SetData 0, "ABC-123"
ClientX1.PrinterNumber = 2
ClientX1.PrintJob
```

## PrinterPath Property

**Important:** This property has been deprecated. It is included in the Software Print Server for backward compatibility.

### Syntax

```
printerPath = Object.PrinterPath
Type = string, read/write if not using ClientIniPath, otherwise, read only
```

### Description

The PrinterPath reflects the path to the LPS install directory or the path that contains the print32.ini file. The default LPS path on the server path is:

32-bit: C:\Program Files\Software Labeling

64-bit: C:\Program Files (x86)\Software Labeling

This property MUST be set in order to obtain the printer information (PrinterCount, PrinterName, PrinterPort, or PrinterAlias)

**Example**

```
' Set the PrinterPath for the Default server (0)
ClientX1.PrinterPath = "\\XFILES\SOFTWARE"
```

## PasExt Property

**Important:** This property has been deprecated. It is included for backward compatibility.

### Syntax

```
pasExt = Object.PrinterPath
Type = string, read/write if not using ClientIniPath, otherwise, read only
```

## Description

The PasExt Property reflects the extension that the LPS is expecting PAS files to use. If files are not processed in the LPS scan directory, it may be because the server is looking for files of a different extension. The default PAS file extension is .pas which is the default for the server as well. The "." should not be specified when setting this property.

### Example

```
' Set the PasExt for the Default server (0)
' This is for non-standard extensions only
ClientX1.PasExt = "pvv"
```

## Quantity Property

### Syntax

```
Object.Quantity
Type = short, read/write
```

### Description

The Quantity property is a read/write property that sets the amount of labels to print. The default for this property is one. The Quantity of labels is used to increment serial numbers and other incrementing fields.

When using this property with a label that has an incrementing or decrementing serial number, it is important to note that changing this property DOES NOT create duplicate serial numbers. To create multiple copies of labels with identical serial numbers use the Duplicates Property.

### Example

```
'This example ends up producing 10 labels
'with 2 duplicates of any incrementing or decrementing number
ClientX1.SetLabelName "Label1.lwl"
ClientX1.SetData 0, "ABC-123"
ClientX1.Quantity = 5
ClientX1.Duplicates = 2
ClientX1.PrintJob
```

## ServerCount Property

### Syntax

```
ServerCount = Object.ServerCount
Type = short, read only
```

### Description

ServerCount is used along with ServerNumber, ServerName, ServerAlias, and ClientIniPath properties. ServerCount only reflects a value greater than 1 (one) if a valid ClientIniPath & configuration file (LLMWCInt.ini) with multiple servers listed is set. The default for this property is 1 (the Default

Server). This property is useful for iterating through a list of servers to get the names and aliases for display to the user. If you are not using the ClientIniPath property with multiple servers specified in the .ini file, ServerCount has no use.

#### Example

'this example sets the ClientIniPath property to a .ini file name. It then  
'populates a list box control with the server names and aliases found in  
'the ini file. The iteration is controlled with the ServerCount property.

```
Dim i As Integer
On Error GoTo Handler
frmFront.ClientX1.ClientIniPath = Trim(txtClientIniLocation.Text)
For i = 1 To frmFront.ClientX1.ServerCount - 1
    frmFront.ClientX1.SetServer (i)
    lstServers.AddItem frmFront.ClientX1.ServerName & " ALIAS " &
    frmFront.ClientX1.ServerAlias
Next i
```

## ServerNumber Property

### Syntax

```
serverNumber = Object.ServerNumber
Type = short, read only
```

### Description

ServerNumber is used along with ServerCount, ServerName, and ServerAlias properties. ServerNumber reflects the index of the internal Server array. The default for this property is 0. This property enables you to retrieve, in code, the number of the currently selected server that was selected with the SetServer Method. If you are not using the ClientIniPath property with multiple servers specified in the .ini file, ServerNumber has no use.

#### Example

'this example gets the currently selected server and sets the system to the  
'next one. If the next one goes beyond server count, 0 is used  
'(the default server)

```
Dim I as integer
serverNumber = ClientX1.ServerNumber
' Now let us get the next server
serverNumber = serverNumber + 1
If (serverNumber = ClientX1.ServerCount) then
    serverNumber = 0
end if
ClientX1.SetServer(serverNumber)
```

## ServerName Property

### Syntax

```
serverName = Object.ServerName
Type = string, read only
```

### Description

ServerName is used along with ServerCount, ServerNumber, and ServerAlias properties. ServerName reflects the name of the currently selected Server. If you are not using the ClientIniPath property with multiple servers specified in the .ini file, ServerName has no use.

#### Example

```
'this example sets the ClientIniPath property to a .ini file name. It then
'populates a list box control with the server names and aliases found in
'the ini file. The iteration is controlled with the ServerCount property.
```

```
Dim i As Integer
On Error GoTo Handler
frmFront.ClientX1.ClientIniPath = Trim(txtClientIniLocation.Text)
For i = 1 To frmFront.ClientX1.ServerCount - 1
    frmFront.ClientX1.SetServer (i)
    lstServers.AddItem frmFront.ClientX1.ServerName & " ALIAS " &
    frmFront.ClientX1.ServerAlias
Next i
```

## ServerAlias Property

### Syntax

```
serverAlias = Object.ServerAlias
Type = string, read only
```

### Description

ServerAlias is used along with ServerCount, ServerNumber, and ServerName properties. ServerAlias reflects the alias name of the currently selected Server. If you are not using the ClientIniPath property with multiple servers specified in the .ini file, ServerAlias has no use.

#### Example

```
'this example sets the ClientIniPath property to a .ini file name. It then
'populates a list box control with the server names and aliases found in
'the ini file. The iteration is controlled with the ServerCount property.
```

```
Dim i As Integer
On Error GoTo Handler
frmFront.ClientX1.ClientIniPath = Trim(txtClientIniLocation.Text)
For i = 1 To frmFront.ClientX1.ServerCount - 1
    frmFront.ClientX1.SetServer (i)
    lstServers.AddItem frmFront.ClientX1.ServerName & " ALIAS " &
    frmFront.ClientX1.ServerAlias
Next i
```

## TrimLeadingSpaces Property

### Syntax

```
Object.TrimLeadingSpaces = Boolean
Type = short, read/write
```

### Description

The TrimLeadingSpaces property trims the leading spaces from data before it presented to the label. If your program has control of the data, you should do any trimming or data manipulation before using the "SetData" Method. If you are going to use this property, it must be set before using the SetData Method. The default for this property is FALSE.

#### Example

```
'This example ends up producing 10 labels
'The spaces before "ABC" are removed by the system
ClientX1.SetLabelName "Label1.lwl"
ClientX1.TrimLeadingSpaces=TRUE
ClientX1.SetData 0, "  ABC-123"
ClientX1.PrintJob
```

## ActiveX Client Control Methods

### AppendJob Method

#### Syntax

```
Object.AppendJob
```

#### Description

AppendJob is used to queue up the current label that has been designated via the SetData method calls, as well as Quantity, Duplicates, Pages, and/or PrinterNumber properties. AppendJob is used for batching label requests together for one server request, instead of many separate jobs. The resulting .pas file that is placed in the DropDirectory contains a "stacked" list of all requested labels.

**Note:** If the PrinterNumber property is not set before invoking this method, the default printer (Printer 1) is used.

**Example**

```
'Queue up several labels and then print

For j = 1 To numberOfLabels
    ClientX1.ClearData
    populateControlwithRandomData
    ClientX1.Quantity = MaskQuantity.Text
    ClientX1.Duplicates = maskDuplicates.Text
    ClientX1.printerNumber = printerNumber
    ClientX1.AppendJob
Next j
'the batch is now complete, submit the job to LPS
ClientX1.PrintJob
```

## ClearData Method

### Syntax

```
Object.ClearData
```

### Description

ClearData is used to clear all the data members for the current label, ensuring there are no ‘sticky’ data values being sent for the next job.

**Example**

```
'Queue up several labels and then print
For j = 1 To numberOfLabels
    ClientX1.ClearData
    populateControlwithRandomData
    ClientX1.Quantity = MaskQuantity.Text
    ClientX1.Duplicates = maskDuplicates.Text
    ClientX1.printerNumber = printerNumber
    ClientX1.AppendJob
Next j
' The Batch is complete, now send it
ClientX1.PrintJob
```

## Login Method

### Syntax

```
Object.Login (IPAddress as string, ServerName as string)
```

### Description

The Login method is the fastest and easiest way to login to a single LPS through a socket connection. When called, the ActiveX Client Control issues a login request to the LPS. The LPS must be online and started . The LPS then responds and opens a communication socket between itself and the application using the ActiveX Client Control. Once connected, information is sent back and forth between the client and the server, including any properties or methods used during the current session.

**Example**

```
'use the Login method connect to connect to a single LPS
'The following issues a Login Request on an LPS running on a server named
'SHIPLINE1 with an IP Address of 165.10.0.120
ClientX1.Login "165.10.0.120", "SHIPLINE1"
```

## PrintJob Method

### Syntax

```
Object.PrintJob
```

### Description

PrintJob is used to submit the current job to the LPS. The current data is supplied via the SetData method calls, as well as Quantity, Duplicates, Pages, and/or PrinterNumber properties. The job is submitted to LPS.

**Example**

```
'initialize the control to point to the LPS
'assume softwarePath has been preset to the network location of
'the Software print server (LPS). GScanDirectory has been preset
'to the directory where the LPS is scanning.
ClientX1.LabelsPath = softwarePath & "\\labels"
ClientX1.LayoutPath = softwarePath & "\\layouts"
ClientX1.DropDirectory = gScanDirectory
ClientX1.PrinterPath = softwarePath
ClientX1.SetLabelName "mytest.LWL"
'populate the data fields with data typed into a text input control array
For i = 0 to ClientX1.FieldCount - 1
    ClientX1.SetData(i) txtfield(i)
Next i
ClientX1.PrintJob
```

## ResetJob Method

### Syntax

```
Object.ResetJob
```

### Description

ResetJob is used to clear the entire job, including any labels that have been appended to the batch with the AppendJob method. This is useful for instances where the end user has changed their mind and wishes to cancel the job.



**Example**

```
' Want to send x number of labels in one job to the printer
dim inner as integer
dim outer as integer
For outer = 0 to x
    'Ensure all the data for the current label is cleared
    ClientX1.ClearData
    for inner = 0 to ClientX1.FieldCount -1
ClientX1.SetData inner, "Some Data"
    Next inner
    'Append this label to the batch
    ClientX1.AppendJob
Next outer
' Did the user click the cancel button?
If cmdCancel.Enabled = FALSE then
    ' Yep, lets u delete all this data
    ClientX1.ResetJob
    Exit function ' And leave
Else
    'The Batch is complete, now send it
    ClientX1.PrintJob
End If
```

## SetData Method

### Syntax

```
Object.SetData FieldName or index as Variant, Data as String
```

### Description

The 'FieldCount', 'FieldName', 'Field Data', and 'FieldLength' properties describe the array that is created when the 'SetLabelName' method is invoked. SetData accesses this array and populates the field specified by either index number or field name with the data specified. Your program may not know ahead of time which and how many fields are in the label. This is why we allow you to iterate through the array with an index. If you do know your field names ahead of time, it is easier to set their data using the 'FieldName' property.

**Note:** You must do a 'SetLabelName' before you can set data using 'SetData'. You do not have to redo a 'SetLabelName' after doing a 'PrintJob'.

**Example 1**

```
'Sending data to the array by using the field's index number:
'setting label data by iterating through field count
ClientX1.SetLabelName "mytest.LWL"
'populate the data fields with data typed into a text input control array
For i = 0 to ClientX1.FieldCount - 1
    ClientX1.SetData(i) txtfield(i)
Next i
ClientX1.PrintJob
```

**Example 2**

'This example builds an SQL statement on the fly based on the field names  
'in the label. The database is then hit with a random key field and the  
'data for the fields is set. See the "Interface" section of Section 1 for  
'more information on providing data for the general case.

**Assumptions**

'The field names in the label format are constrained to the field names in  
'the database. (See Section 1)  
'Global myDatabase As Database defined in module1.  
'Global myRecordset As Recordset defined in module1  
'assume softwarePath has been preset to the network location of the  
'Software Print Server (LPS).  
'GScanDirectory has been preset to the directory where the LPS is scanning.

```
Public Sub populateControlwithRandomData()
Dim sqlStatement As String, i As Integer, thisFieldName As String
'initialize the control to point to the LPS
ClientX1.LabelsPath = softwarePath & "\\labels"
ClientX1.LayoutPath = softwarePath & "\\layouts"
ClientX1.DropDirectory = gScanDirectory
ClientX1.PrinterPath = softwarePath
'open the database
Set myDatabase = OpenDatabase(App.Path & "\\sample.mdb")
'build SQL statement to grab data for this label
sqlStatement = "SELECT "
ClientX1.SetLabelName "mytest.LWL"
For i = 0 To ClientX1.FieldCount - 1
    sqlStatement = sqlStatement & "[" & ClientX1.FieldName(i) & "]" & ", "
Next i
'get rid of the last comma before the FROM clause and append key
sqlStatement = Left(sqlStatement, Len(sqlStatement) - 2) & " FROM Newwar
WHERE NAME1='" & cboRecordChoice.Text & "';"

'grab the record and populate the data
'only grab the fields we need for this label from the database
Err = 0
On Error Resume Next
Set myRecordset = myDatabase.OpenRecordset(sqlStatement)
If Err <> 0 Then
    MsgBox "SQL Error #" & Err & " SQL = " & sqlStatement, vbInformation, "SQL Error"
    Exit Sub
End If
'populate the label fields with the retrieved data
For i = 0 To ClientX1.FieldCount - 1
    thisFieldName = ClientX1.FieldName(i)
    ClientX1.SetData thisFieldName, myRecordset.Fields(thisFieldName)
Next i
myRecordset.Close
pickRandomRecord
myDatabase.Close
'print the label
ClientX1.PrintJob
End Sub
```

**Example 3**

```
'setting data by iterating through field count instead of using field name
'this example sets every field to "sample data"
ClientX1.SetLabelName "ibm.LWL"
For I = 0 to ClientX1.FieldCount - 1
    ClientX1.SetData I, "sample data"
Next I
ClientX1.PrintJob
```

## SetLabelName Method

### Syntax

```
Object.SetLabelName (newLabelName as String)
```

### Description

The SetLabelName method retrieves the field information for the specified label format. This method also creates an array to hold the variable data for every variable field on the current label format. This array is iterated by index number or field name. If you do not know the field names ahead of time, you must use an index into the array. See 'FieldCount' and 'SetData' for more information on this.

The SetLabelName Method does the following: Creates an Array to hold label field names, maximum field lengths, and data.

**Note:** Because the array is contained within the control, it is Private. The only way to get information to and from the array is through the control's properties and methods. The Array that is created has the following structure:

```
FieldName() as a String
FieldLength() as a short
FieldData() as a String
```

The first index number of the array is zero.

**Example**

```
'assume softwarePath has been preset to the network location of the
'Software print server (LPS). GScanDirectory has been preset to the
'directory where the LPS is scanning.
ClientX1.LabelsPath = softwarePath & "\\labels"
ClientX1.LayoutPath = softwarePath & "\\layouts"
ClientX1.DropDirectory = gScanDirectory
ClientX1.PrinterPath = softwarePath
ClientX1.SetLabelName "AIAG.LWL"
ClientX1.SetData "PARTNUMBER", "A100"
ClientX1.PrintJob
```

## SetServer Method

### Syntax

```
Object.SetServer ServerName or index as Variant
```

## Description

SetServer is only useful if you are using the ClientIniPath property to specify multiple Loftware servers in the llmwclnt.ini file. After pointing the ClientIniPath property to an .ini file that contains references to multiple Loftware servers, you may use the SetServer method to tell the control which server to use. The Labels, Layouts, and Printers paths automatically inherit the values specified in the corresponding entry in the .ini file. If you need to disable a previous use of the ClientIniPath property, "SetServer 0" causes the default server to be selected. That uses the "LabelsPath," "LayoutPath," "DropDirectory," and "PrinterPath" properties manually set with the control instead of the ones that are specified in the .ini file. Therefore, server numbers specified in the .ini file start at 1.

### Example

```
'this example sets the ClientIniPath property to a .ini file name. It then
'populates a list box control with the server names and alias's found in
'the ini file. The iteration is controlled with the ServerCount property, with
'the selected server being controlled with the SetServer Method. Notice the
'default server (0) is not included in the loop.
```

```
Dim i As Integer
On Error GoTo Handler
frmFront.ClientX1.ClientIniPath = Trim(txtClientIniLocation.Text)
For i = 1 To frmFront.ClientX1.ServerCount - 1
    frmFront.ClientX1.SetServer (i)
    lstServers.AddItem frmFront.ClientX1.ServerName & " ALIAS " &
    frmFront.ClientX1.ServerAlias
Next i
```

## ActiveX Client Control Events

### ErrorEvent Event

#### Syntax

```
errorID as Long, errorString as String
```

#### Description

Passes error messages back to the container during various operations. ErrorEvent items are critical and indicate that an error is generated (caught by error handling). An operation might send multiple error events, but the first occurrence of the ErrorEvent is the error that is passed back to the container to the error handling routines. See the ErrorEventIDs table for a breakdown of all warning events that are generated. Make sure that your program takes appropriate action after receiving one of these events. The safest bet when it comes to the error event is to end the program.

**Note:** It is critical that you trap the error event. This is especially true if the LPS server is running in a clustered environment and a failover occurs. Many of the ClientX methods throw critical errors during the failover transition. The following error trap handles this scenario:

**Example**

```
'display a message box showing the error number and string
Private Sub ClientXl_ErrorEvent(ByVal errorID As Long, ByVal errorString As String)
Dim nResult, szMsg
szMsg = "Error or Failover event -" & vbNewLine
szMsg = szMsg & "Software ActiveX Error Event #" & errorID & " has occurred. Error
String = " & Trim(errorString) & ". Consult the documentation for the Software
ActiveX Client control for more information." & vbNewLine & vbNewLine
szMsg = szMsg & "Cluster Failover -" & vbNewLine
szMsg = szMsg & "If your Software Print Server is running in a clustered
environment, you may be experiencing a failover situation. Wait 30 seconds and try
your program again."
nResult = MsgBox(szMsg, vbCritical, "Software - Critical Error or Failover in
Progress")
End
End Sub
```

**ErrorEvent Ids and Messages**

Also includes error returned from the control.

ID	Message	Explanation
25500	Name not found!	Name entry in llmwcint.ini file not found.
25501	PrinterPath not found!	PnnterPath entry in llmwcInt.ini file not found
25502	Unable to locate path to Printr32.ini file!	While attempting to obtain the printer information, the control could not locate the path for the printer information.
25503	Critical Failure of job # jobNum (Not currently used in the control)	A critical failure of a job has been reported
25504	Error on printer prnter number	The server is reporting a printer error on the listed printer. (Not currently used in the control)
25505	Printer Name: Not Found	The printer name was not located in the printr32.ini file
25506	Printer Port: Not Found	The printer port was not located in the printr32.ini file.
25507	Printer ID: Not Found	The printer id was not located in the printr32.ini file.
25508	No servers located. Application cannot print labels!	There were no server entries (sections) located in the llmwcInt.ini file.
25510	Redirected Ini File does not exist!	The redirected ini file was not located.

ID	Message	Explanation
25512	Failed to obtain the Computer Name!	The computer name is required to generate the unique jobname as well as unique filenames.
25513	Path does not exist!	Tested path does not exist!
25514	Write Access Denied!	Write access (For the DropDirectory) has been denied during the test.
25515	Read Access Denied for File: file	Cannot read file in Label, Layout, or PrinterPath.
25516	File does not exist!	Tested file does not exist (printr32.ini).
25517	Read Access Denied!	Read access for a given directory is denied (LabelsPath, LayoutPath, or PrinterPath).
25518	Sever name is stopped	Recorded during isRunning call if server is not running. (Yet accessible)
25519	Server name is unreachable!	Recorded during isRunning call if the DropDirectory is not set, or is incorrect.
26500	Unable to open file.	Unable to open the label file (unlikely)
26501	Unable to open file 'file'.	Unable to open the label file
26502	Unable to change file mode for 'file'	Internal error when opening and parsing the label file. (Try again)
26503	Tab marker information not located in file 'file'	The requested label file has been saved under a previous version.
26504	Label not set	An attempt to access a field or print a job with having called SetLabelName() first.
26505	Not a valid index number	An invalid index number. (<0 or greater than the count)
26506	Fieldname 'name' not found	A call to SetData or Fieldlength with the invalid fieldname 'name'.
26507	Printerpath is not set	An attempt was made to access printer information without first setting the PrinterPath or ClientiniPath, or the path was determined invalid.
26508	DropDirectory is not set.	An attempt to call Printjob without setting the DropDirectory or ClientiniPath. Or the path was determined invalid
26509	Not a Valid Printer Number	A negative printer number was passed
26510	Printeralias 'alias' not found	An invalid printer alias was passed to PrinterPort.

ID	Message	Explanation
26511	Error creating pass file 'file'.	There was an error while attempting to create a PAS file in the DropDirectory.
26512	ClientIni file 'file' not found	The llmwcInt.ini file was not located in the specified path (ClientIniPath).
26513	No sections located in ini file.	There were no server sections located in the llmwcInt.ini file.
26514	Value cannot be negative.	Quantity, Duplicates, Pages, or PrinterNumber was set to a negative number.
26515	Could not determine the fieldname/position	Call to SetData with a zero-length string for a fieldname.
26516	Invalid Server Name	SetServer was passed a ServerName that does not exist.
26517	Invalid Server Number	SetServer was passed a server number or index that does not exist.
25520		Printer Error ...job should print when error is cleared.
25261		All requests failed (Critical)
25262		Stacked job partially printed
27000		Ignore

## InfoEvent Event

### Syntax

```
infoID as Long, infoString as String
```

### Description

Passes informational messages back to the container during various operations. InfoEvent items are used mainly as checkpoints and can be useful if displayed to the end user during initialization, etc. See the InfoEvent ID table for a breakdown of all infoevents that may be generated. This is a useful event during the development/debug process of your program. Once your program is fully debugged, you do not have to pay much attention to this event.

#### Example

```
Private Sub ClientX1_InfoEvent(ByVal infoID As Long, ByVal infoString As String)
' Log these messages to an info multi-line textbox
lstInfoList.AddItem infoString
End sub
```

## InfoEvent IDs and Messages

ID	Message	Explanation
25000	LabelsPath: path	Displays the LabelsPath.
25001	LayoutPath: path	Displays the Layout Path.
25002	PrinterPath: path	Displays the PrinterPath.
25003	ScanPath: path	Displays the DropDirectory.
25004	StatusPath: path	Displays the internal path used for status.
25005	PasExt: ext	Displays the extension used for PAS files.
25006	Name: name	Displays the ServerName.
25007	Alias: alias	Displays the ServerAlias.
25008	Gathering Printer Information	Checkpoint before obtaining the printer info.
25009	Printer number	Displays the PrinterNumber
25010	Printer name:	Displays the PrinterName
25011	Printer Port: port	Displays the PrinterPort
25012	Printer ID: number	Displays the printer ID (Not used)
25013	Printer Alias: alias	Displays the PrinterAlias
25016	Looking for ini File: file	Checking for the existence of the LLMWCInt.ini file in the ClientIniPath.
25017	Using Redirected ini File: file	Detected the use of a redirected LLMWCInt.ini file and attempts to obtain config information from the listed file.
25018	Getting information for entry: section	Lists the current section in the ini file being processed.
25019	Verified Path Exists	Path existence test OK.
25020	Verified Write Access	Path write test OK
25021	Verified Existence of type files	Files with extension of type were found in directory
25022	Verified Read Access for one file: file	Verified the ability to open a file for read (filename is displayed).
25023	Verified Read Access	Verified read access to a directory
25025	Verified File Exists	A file has been checked for existence
25029		Job Printed



## WarningEvent Event

### Syntax

```
warningID as Long, warningString as String
```

### Description

Passes warning messages back to the container during various operations. WarningEvent items are used to flag a missing item, etc., which is not detrimental to the operation, but is out of the ordinary. See the WarningEventID table for a breakdown of all warning events that may be generated.

#### Example

```
'display a message box showing the error number and string
Private Sub ClientX1_WarningEvent(ByVal warningID As Long, ByVal warningString As String)
MsgBox("Warning # " & warningID & " has occurred. Message = " & warningString)
End sub
```

### WarningEvent IDs and Messages

ID	Message	Explanation
25250	Alias Not Found!	Server Alias was not located (optional entry) in the llmwcInt.ini file.
25251	LabelsPath not found	There was no entry for LabelsPath in the llmwcInt.ini file, the default is the "Labels" sub-directory off the PrinterPath.
25252	LayoutPath not found	There was no entry for LayoutPath in the llmwcInt.ini file, the default is the "Layouts" sub-directory off the PrinterPath.
25253	ScanPath not found	There was no entry for ScanPath in the llmwcInt.ini file, the default is the "WDDrop" sub-directory off the PrinterPath
25254	PrinterAlias: Not Found	There was no Printer Alias defined for the particular printer.
25256	Could not locate any 'type' files	No files of type "type" were located when checking the path.
25257	Unable to test Read Access!	Since there were no files located, read access on files cannot be checked.
25258	Failed to obtain the Username of the currently logged in user	The username of the current logged in user is unable to be determined (used for job tracking, on the back end).

## OtherEvent Event

### Syntax

```
otherID as Long, otherString as String
```

## Description

Other events are not currently used, but they contain information that does not fit into the info, warning, or error categories.

### Example

```
Private Sub ClientX1_OtherEvent(ByVal otherID As Long, ByVal otherString As String)
' Log these messages to an info multi-line textbox
txtOther.Text = otherString
' No ID tests as there are no known other Ids. . .
End sub
```

## ActiveX Client Control Properties and Methods

Name	Read	Write	Type	Default	Comment
<a href="#">AppendJob</a>	N/A	N/A	Method	N/A	Batches label requests together.
<a href="#">ClearData</a>	N/A	N/A	Method	N/A	Clears all the data members for the current label
<a href="#">ClientIniPath</a>	X	X	Property	N/A	Allows use of LLMWCInt.ini file for the control to read the properties from the LPS, or to connect to multiple .servers through socket connections.
<a href="#">DropDirectory</a>	X	X*	Property	N/A	Reflects the path to the server's LPS scan path.
<a href="#">Duplicates</a>	X	X	Property	1	Sets the amount of duplicate labels to print.
<a href="#">ErrorEvent</a>	N/A	N/A	Event	N/A	Passes error messages back to the container.
<a href="#">FieldCount</a>	X	X	Property	N/A	Displays how many variable fields there are in the current label format
<a href="#">FieldLength</a>	X	N/A	Property	N/A	Displays the length of a specified field in the current label format
<a href="#">FieldName</a>	X	X	Property	N/A	Displays the name of a specific field in the current label format.
<a href="#">InfoEvent</a>	N/A	N/A	Event	N/A	Passes informational messages back to the container.
<a href="#">isRunning</a>	N/A	N/A	Property	N/A	Checks to see if the selected LPS is started.
<a href="#">JobName</a>	X	X	Property	Contains computer name	Reflects a unique identifier for the current job.

<b>Name</b>	<b>Read</b>	<b>Write</b>	<b>Type</b>	<b>Default</b>	<b>Comment</b>
<u>LabelsPath</u>	X	X*	Property	N/A	Reflects the UNC or mapped drive path to the labels directory used by the LPS.
<u>LayoutPath</u>	X	X*	Property	N/A	Reflects the UNC or mapped drive path to the LPS's Layout Directory.
<u>Login</u>	N/A	N/A	Method	N/A	Used to log in to a single LPS through a socket connection.
<u>OtherEvent</u>	N/A	N/A	Event	N/A	Contains information that does not fit into the info, warning or error categories
<u>Pages</u>	X	X	Property	1	Sets how many pages of labels are printed.
<u>PrinterAlias</u>	X		Property	N/A	The descriptive name for an assigned printer.
<u>PrinterCount</u>	X		Property	N/A	The number of configured printers.
<u>PrintJob</u>	N/A	N/A	Method	N/A	Used to submit the current job to the LPS.
<u>PrinterName</u>	X		Property	N/A	Name of configured printer.
<u>PrinterPort</u>	X		Property	N/A	List of configured printer ports
<u>PrinterNumber</u>	X	X	Property	1	Corresponds to configured Printers
<u>PrinterPath</u>	X	X*	Property	N/A	Reflects the path to the LPS install directory.
<u>PasExt</u>	X	X*	Property	.pas	Reflects the extension that the LPS is expecting PAS files to use.
<u>Quantity</u>	X	X	Property	1	Sets the amount of labels to print.
<u>ResetJob</u>	N/A	N/A	Method	N/A	Clears the entire job, including appended labels.
<u>ServerCount</u>	X		Property	N/A	Iterates through a list of servers to display names and aliases to the user.
<u>ServerNumber</u>	X		Property	N/A	Number of currently selected server.

<b>Name</b>	<b>Read</b>	<b>Write</b>	<b>Type</b>	<b>Default</b>	<b>Comment</b>
<u>ServerName</u>	X		Property	N/A	The name of the currently selected server.
<u>ServerAlias</u>	X		Property	N/A	The alias name of the currently selected server.
<u>SetData</u>	N/A	N/A	Method	N/A	Populates the field by index number or field name.
<u>SetLabelName</u>	N/A	N/A	Method	N/A	Retrieves field information for the specified label format.
<u>SetServer</u>	N/A	N/A	Method	N/A	Tells the control which server to use.
<u>TrimLeadingSpaces</u>			Property	0	Trims the leading spaces from the data before it is presented to the label.
<u>WarningEvent</u>			Event	N/A	Passes warning messages back to the container during various operations.

\*Only if not using ClientIniPath

This page intentionally left blank

Like the WebClient, the Internet ActiveX, or “iX” as it is known, prints across the Internet to locally selected printers that have been configured as CLIENT DEFINED on the server. This section contains references to applications that are also documented in previous sections. It is recommended that you read the Internet Printing section, before continuing.

This section includes information on the use of Web Servers, JSPs, servlets, and servlet engines, all of which are components of the Internet ActiveX.

Programmers developing in 32-bit languages supporting ActiveX Controls can easily interface the Software Print Server functions with their own applications. In the previous ActiveX sections, Software discussed the Client Control. These controls may be employed with ease by a user to connect and print to stand-alone and networked barcode label printers. The Internet ActiveX Control (“iX”) can utilize this innovative technology as well.

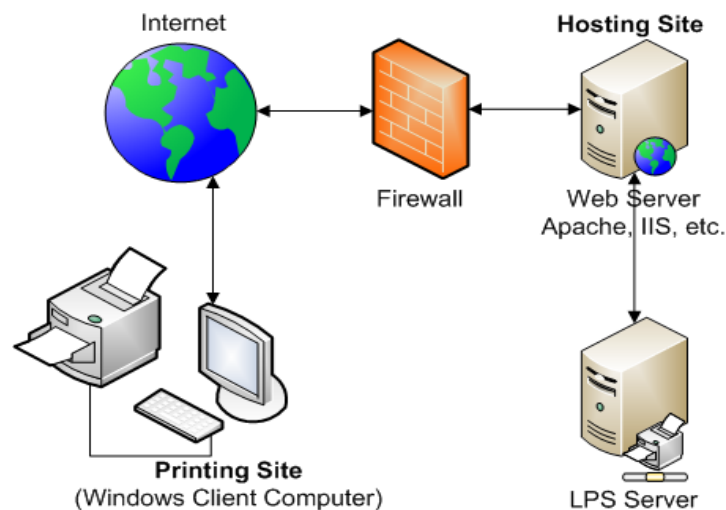


Figure 6-A: The Concept of Internet ActiveX

This section documents the Software Internet ActiveX Control. Like the WebClient and Software's other ActiveX Client Controls, iX has a “thin” footprint because it does not require that the Software Label Manager subsystem be installed on the same computer as the control. It is called the “Internet ActiveX” because it acts as a client across the Internet to the LPS (Software Print Server). Use this control when your application is running in several places, needs to access many printers, and requires a small footprint.

## Installation/Use of the Internet ActiveX Control

### Pre-Install Checklist

Have you installed the following?

- Software Print Server

**Note:** The Software Print Server can be installed on any computer with a supported operating system anywhere on your LAN or WAN.

- Web Server
- Servlet Engine
- LPS Web Servlet
- Client-defined Printers

**Hint:** Before proceeding, install and run the WebClient from a Client computer to ensure that the Software Print Server installation is successful .

### Installing the Internet ActiveX Control

Install the Internet ActiveX Control to the Client computer one of the following ways:

- A Full install from the CD adds the web x dll.
- A Client install from the LPS Client Install Folder on the CD.
- An Internet ActiveX Control install, either from the CD, or hosted on the Web Server and downloaded by the Client computer.

### Using the Internet ActiveX Control

By running the sample program, called Internet ActiveX Sample 1, and examining the code, you will be able to see how each property, method and event of the Internet ActiveX control is used. Sample programs are installed to Software Labeling\Sample Programs with the Software Print Server Premier Edition.

The sample program can be run from the computer where Software is installed or a client computer. To run the program from a client computer, install the Software Print Server WebClient from the LPS Client Install folder of the Software CD. Before opening the sample program:

- Configure the connected printer as CLIENT DEFINED (using Software Label Manager) on the server.
- Design a label format in Software Label Manager. Make sure the label you design prints successfully from Software Label Manager.

Once you complete the previous steps, you can open the Internet Active X Control Sample 1 program from the Software Labeling\Sample Programs folder.

- Open on InternetActiveControlSample1.exe.
- Open the .vbp file to open the program in Visual Basic, if Visual Basic is installed on your computer.

**Software's Internet Active X Control Sample 1**

**Software's iXControl**

**ATTENTION:** Items in RED are required fields in order to connect and print a label through the web server. Items in BLUE are the Methods, Properties or Events used with the associated object.

**STEP 1** Enter the following data:

\* Enter URL to Web Server \* Enter the Port

Enter the User Name Enter the Password

User Name and Password are optional. They are used if security is needed for access to the Internet

**STEP 2** Press "Connect" to connect to the Web Server

\* Connect CWebX1.Connect (Call Statement)

Disconnect CWebX1.Disconnect

Connection Successful? CWebX1.isConnected

**STEP 3** Press the "Get Printers" button to return a list of configured printers on the LPS

Get Printers CWebX1.PrinterAlias CWebX1.PrinterName CWebX1.PrinterPort CWebX1.PrinterCount

**STEP 4** Enter the Printer Number from the list in step 3 and the Printer Port the label will be printed to on this computer. Press the "Set Printer" button.

Printer Number: \* 1 \* Set Printer CWebX1.SetPrinter (Call Statement)

Printer Port: \* COM2 \* Press the "Test Print" button to eject a label from the printer to test communication with the printer.

Printer Timeout: \* 8 \* Test Print CWebX1.TestPrinter (Call Statement)

**STEP 5** Enter the label name to print and press "Set Label Name"

LabelName: \* [Text Box] \* Set Label Name CWebX1.SetLabelName

☒ Trim Leading Spaces CWebX1.TrimLeadingSpaces

**STEP 6** Highlight a field and enter data to print for the highlighted field in the "Field Data" text box and press "Set Data".

[List Box] CWebX1.FieldCount CWebX1.FieldName CWebX1.FieldLength

Field Name: [Text Box]

Field Data: [Text Box]

Click here to queue the present job, then enter more data to print. Append Job CWebX1.AppendJob

\* Set Data CWebX1.SetData

**STEP 7** Enter the Quantity of labels to print and press "Print Job"

\* Quantity Duplicates 1 1 \* CWebX1.Quantity CWebX1.SetDuplicates

Cancel Print Job CWebX1.ClearAllData CWebX1.ResetJob

\* Print Job CWebX1.PrintJob

Figure 10.1: Internet ActiveX Sample Program

Follow the steps that are displayed on the form to print a label. The Blue items are the associated properties, methods or events associated with the corresponding object. Items with Red asterisk are required fields.

1. Create a project in a programming tool that can utilize ActiveX Controls.
2. Add the Software Internet ActiveX Control to the new project.
3. Connect to the Web Server by invoking the Connect Method.
4. Select a printer by invoking the SetPrinter Method.
5. Choose a label by invoking the SetLabelName Method.
6. Set field data by using the SetData Method.
7. Print the label format by using the PrintJob Method.

### Related Information

For instructions on how to install the Software Print Server, refer to *The Software Print Server* section of this guide



For information on installing the Web Server, Servlet Engine, and LPS Web Servlet, refer to the *Internet Printing* section of this guide.

For information on configuring client-defined printers, refer to the Device Connections section of the *Software Label Manager User's Guide*.

## Troubleshooting the Internet ActiveX Client Control

**Important:** It is critical that you trap error events. This is especially true if the LPS server is running in a clustered environment and a failover occurs. Many of the iX methods throw critical errors during the failover transition.

- The control uses a list inside the label format that is generated when the label is saved. Error Event #26503 is thrown if the label format does not contain the list.
- Make sure that you have some labels designed and printers configured on the LPS as CLIENT DEFINED before trying to use the PrintJob Method.
- Make sure you trap errors and display the error string in the ErrorEvent. This saves considerable debug time.
- You may want to add a multi-line list box to your application that you can add a line to for each WarningEvent and InfoEvent. This provides valuable information that helps you get up and running quickly. You can remove these events when they are no longer needed.

### Related Information

For more information on iX errors, refer to the Internet ActiveX *Error Event* section in this guide.

## Design Scenario and Distribution

### Overview

The iX control can be integrated into the user or integrator's applications running on remote systems. The Software Print Server (LPS) is installed at a central location driving the connected clients through the Software Web Servlet component. Printers that are not normally accessible from the central location can be driven with the iX control.

### Design Scenario

Your company has a central site and remote locations not connected via a WAN. This could be as simple as the building next door, a store in the next city, a plant or hub in the next state, or a facility in another country. No matter the scenario, the only [reasonable] connectivity to the site(s) is through the Internet (dial-up, frame, T1, etc.). You have an application that is installed or accessed at the site(s) that has the ability to integrate ActiveX components. The clients that access your application have connectivity to the Internet.

You can include the iX control in your application to allow for printing at each client location. The Software Print Server as well as the Software Web Servlet can be installed at the central location. The iX control only needs to know the address of the central web server, to establish and maintain connectivity. You also need to let the iX control know about the location of the locally accessible printer (s).

The iX control can also be used internally as well. The necessary component is the Software Web Servlet (running on the Web Server). If you are looking for an Intranet solution, the iX control can communicate to the Software Print Server through your internal Web Server. If you already have a WAN and have an intranet application, the iX control can be added to it. Configuration and Operation are the same whether internal or external; it is the location of the Web Server that determines the architecture.

The iX control relies upon the Software Print Server for licensing; you can include the control in your product and distribute it to anyone who requires it. Only when you decide to 'connect' to the LPS will you require a license. This allows integrators and end users the flexibility to include the functionality at design time.

## Distributing the iX Control with your Application

As mentioned previously, the iX Control can be distributed to anyone who requires it. Licensing constraints are handled by the LPS that the Control connects to. When building an install program for your application, include the required Software files with it. This way, your setup program takes care of installing the Software control and its associated files, thereby avoiding having to run two setup programs. The following files are needed to use the control. These files are found in the ...Internet ActiveX/redist directory on the Software CD. If your programming language has a setup wizard, it probably picks up the correct files.

WebX.tlb -> winsys folder

WebX.dll (the control) -> winsys folder. This file needs to be registered.

## Internet ActiveX Properties

Software's ActiveX Properties are as follows:

### Duplicates Property

#### Syntax

```
Object.Duplicates
Type = short, read/write
```

#### Description

The Duplicates property is a read/write property that sets the amount of duplicate labels to print. The default for this property is one. Duplicate labels are EXACT copies of the original label. When using this property with a label that has an incrementing or decrementing serial number, this many labels print

before incrementing (or decrementing) the number. To create multiple labels with unique serial numbers, use the Quantity Property.

#### Example

```
'print 2 copies of each serial number 5 times.
'the total number of labels printed = 10
CWebX1.SetLabelName "WebSerial.lwl"
CWebX1.SetData 0, "ABC-123"
CWebX1.Quantity = 5
CWebX1.Duplicates = 2
CWebX1.PrintJob
```

## FieldCount Property

### Syntax

```
Object.FieldCount
Type = short, read only
```

### Description

The FieldCount property describes the array created when the SetLabelName method is invoked. It is important to understand that this array is the key to having access to all fields in your label in a dynamic fashion. Fields in the array are accessed by their index number in the array or by the field name itself. Your program may not know ahead of time which and how many fields are in the label. This is why you to iterate through the array with an index. If you do know your field names ahead of time, it is easier to set their data using the 'FieldName' property.

The FieldCount property is a read only property that displays how many variable fields there are in the current label format. This allows you to iterate through the field array that is generated when the 'SetLabelName' method is invoked. There is no default for this property.

#### Example

```
Dim i As Integer
Dim fn As String
Dim fs As Integer
Dim line As String

For i = 0 To CWebX1.FieldCount - 1
    Err.Clear
    fn = CWebX1.FieldName(i)
    If (Err.Number <> 0) Then
        line = i & " - Error"
    Else
        fs = CWebX1.FieldLength(i)
        line = i & " - " & fn & " Len: " & fs
    End If
```

## FieldLength Property

### Syntax

```
Object.FieldLength (index) or (FieldName)
Type = short, read only
```

### Description

The 'FieldCount', 'FieldName' and 'FieldLength' properties describe the array created when the 'SetLabelName' method is invoked. It is important to understand that this array is the key to having access to all fields in your label in a dynamic fashion. Fields in the array can be accessed by their index number in the array or by the field name itself. Your program may not know ahead of time which and how many fields are in the label. This is why we allow you to iterate through the array with an index. If you do know your field names ahead of time, it is easier to set their data using the 'FieldName' property.

The FieldLength property is a read only property that displays the length of a specified field in the current label format. This property can be retrieved by the actual field name or field index number. This property is not filled until the SetLabelName method has been invoked.

**Note:** This property is very useful for validating data before actual printing to assure the number of characters supplied for the field does not exceed its Field Length.

There is no default for this property.

#### Example

```
Dim i As Integer
Dim fn As String
Dim fs As Integer
Dim line As String

For i = 0 To CWebX1.FieldCount - 1
    Err.Clear
    fn = CWebX1.FieldName(i)
    If (Err.Number <> 0) Then
        line = i & " - Error"
    Else
        fs = CWebX1.FieldLength(i)
        line = i & " - " & fn & " Len: " & fs
    End If
```

## FieldName Property

### Syntax

```
Object.FieldName ( index )
Type = short, read only
```

## Description

The 'FieldName' property is created when the 'SetLabelName' method is invoked. It is important to understand that this array is the key to having access to all fields in your label in a dynamic fashion. Fields in the array can be accessed by their index number in the array or by the field name itself. Your program may not know ahead of time which and how many fields are in the label. This is why we allow you to iterate through the array with an index. If you do know your field names ahead of time, it is easier to set their data using the 'FieldName' property.

The FieldName property is a read only property that displays the name of a specific field in the current label format. This property can only be retrieved by the field index number. This property is not filled until the SetLabelName method has been invoked. There is no default for this property.

### Example

```
Dim i As Integer
Dim fn As String
Dim fs As Integer
Dim line As String
For i = 0 To CWebX1.FieldCount - 1
    Err.Clear
    fn = CWebX1.FieldName(i)
    If (Err.Number <> 0) Then
        line = i & " - Error"
    Else
        fs = CWebX1.FieldLength(i)
        line = i & " - " & fn & " Len: " & fs
    End If
```

## isConnected Property

### Syntax

```
Object.isConnected
Type = boolean, read only
```

### Description

The isConnected property checks to see if the currently selected Software Print Server is connected. If the LPS is not found, or is not connected, an error is thrown in the ErrorEvent. Make sure that you handle this error in the ErrorEvent, if you do NOT, your program may crash!

### Example

```
If (CWebX1.isConnected = True) Then
    MsgBox("LPS is Successfully Connected")
Else
    MsgBox ("Not Connected")
End If
```

## JobName Property

### Syntax

```
Object.JobName
Type = string, read/write
```

### Description

JobName is a read/write property reflecting a unique identifier for the current job. The default value changes after the PrintJob method is called and follows this naming convention:

ComputerName + "X" + unique instance number + "\_" + YYYYMMDDHHNNSS + serial number per second

If you wish to specify your own JobName, it is up to the creator to ensure its uniqueness. It is through this identifier that future job status is returned. The JobName is also reflected in the Status View console that monitors the status and progress of the jobs processed by the LPS. This property is for feedback purposes only. It is not needed to print labels.

#### Example

```
Dim jobName as string
' Get the current JobName to store for reference
jobName = CWebX1.JobName
```

## Pages Property

### Syntax

```
Object.Pages
Type = short, read/write
```

### Description

The Pages property is a read/write property that sets how many pages of labels are printed. A page of labels is a copy of an entire page of labels created when printing with layouts. The default for this property is one. For more detail on pages of labels, consult the *Software Label Manager User's Guide*. Do not use this property if your label does not use a "multi-up" layout.

#### Example

```
'print 2 identical pages, line break, comment,
'10 labels/page with 2 labels for each serial #

CWebX1.SetLabelName "WebSerial.lwl"
CWebX1.SetData 0, "ABC-123"
CWebX1.Quantity = 5
CWebX1.Duplicates = 2
CWebX1.Pages = 2
CWebX1.PrintJob
```

## PrinterAlias Property

### Syntax

```
Object.PrinterAlias (pindex)
Type = string, read only
```

### Description

Table of printer aliases used in conjunction with: PrinterAlias, PrinterCount, PrinterName, PrinterPort, PrinterNumber, that is populated during Connect().

PrinterAlias is used along with PrinterName, PrinterCount, and PrinterPort properties to allow you to prompt your users for the target-configured printer to which they wish to print. Essentially, these properties expose the list of configured printers that was set up in the Loftware Label Manager. Printer Alias is a descriptive name for a printer that is assigned when in the printer connection dialog box of Loftware Label Manager (same dialog box where you specify a port/spooler/IP address).

#### Example

```
Dim i As Integer
Dim pn As String
Dim pp As String
Dim line As String

For i = 1 To CWebX1.PrinterCount
    pn = CWebX1.PrinterName(i)

    If (Err.Number <> 0) Then
        line = i & " - Not Configured"
    Else
        pp = CWebX1.PrinterPort(i)
        pa = CWebX1.PrinterAlias(i)
        line = i & " - " & "Alias" & " (" & pa & ") " & pn & " on " & pp
    End If

    List1.AddItem (line)
Next
```

## PrinterCount Property

### Syntax

```
Object.PrinterCount
Type = short, read only
```

### Description

PrinterCount is used to return the number of printer seats on the license key. Along with PrinterName, PrinterPort, and PrinterAlias properties, it provides information on the configured printers to which users wish to print. These properties expose the list of configured printers that are set up in the Loftware

Label Manager as CLIENT DEFINED.

#### Example

```
Dim i As Integer
Dim pn As String
Dim pp As String
Dim line As String
For i = 1 To CWebX1.PrinterCount

    Err.Clear

    pn = CWebX1.PrinterName(i)
    If (Err.Number <> 0) Then line = i & " - Not Configured"
Else
    pp = CWebX1.PrinterPort(i)
    pa = CWebX1.PrinterAlias(i)
    line = i & " - " & "Alias" & " (" & pa & ") " & pn & " on " & pp
End If
    List1.AddItem (line)
Next
```

## PrinterName Property

### Syntax

```
Object.PrinterName (pindex)
Type = string, read only
```

### Description

PrinterName allows you to be able to prompt users for the target-configured printer to which they wish to print. Essentially, this property exposes the list of configured printers set up in the Software Label Manager.

#### Example

```
Dim i As Integer
Dim pn As String
Dim pp As String
Dim line As String
For i = 1 To CWebX1.PrinterCount
    line = ""
    Err.Clear
    pn = CWebX1.PrinterName(i)
    If (Err.Number <> 0) Then
        line = i & " - Not Configured"
    Else
        pp = CWebX1.PrinterPort(i)
        pa = CWebX1.PrinterAlias(i)
        line = i & " - " & "Alias" & " (" & pa & ") " & pn & " on " & pp
    End If
    List1.AddItem (line)
Next
```



## PrinterNumber Property

### Syntax

```
Object.PrinterNumber  
Type = string, read/write
```

### Description

Set by application to indicate the printer number for the print job.

Used in conjunction with: PrinterAlias, PrinterCount, PrinterName, PrinterPort, PrinterNumber.

See the “Printer Lists” section.

The PrinterNumber property is a read/write property that sets which configured LPS printer to print to. The default for this property is configured Printer #1. Using the PrinterName or PrinterAlias properties, you could prompt your user with a better description of the target printer. You could then resolve their choice into a number, which ultimately must be used in code to specify the printer.

#### Example

```
Dim P As Integer  
P = txtPrinterNum.Text  
'Set the printer number, printer port and the printer timeout value  
Call CWebX1.SetPrinter(p, txtPrinterPort.Text, txtPrinterTimeout.Text)  
CWebX1.printerNumber = p
```

## PrinterPort Property

### Syntax

```
Object.PrinterPort (pindex)  
Type = string, read only
```

### Description

PrinterPort allows you to be able to prompt your users for the target-configured printer to which they wish to print. They are listed as “Not Configured” or as “CLIENT DEFINED”.

**Example**

```

Dim i As Integer
Dim pn As String
Dim pp As String
Dim line As String
For i = 1 To CWebX1.PrinterCount
    Err.Clear
    pn = CWebX1.PrinterName(i)
    If (Err.Number <> 0) Then
        line = i & " - Not Configured"
    Else
        pp = CWebX1.PrinterPort(i)
        pa = CWebX1.PrinterAlias(i)
        line = i & " - " & "Alias" & " (" & pa & ") " & pn & " on " & pp
    End If
    List1.AddItem (line)
Next

```

## PrinterTimeout Property

### Syntax

```

Object.PrinterTimeout (pindex)
Type = string, read/write

```

### Description

The PrinterTimeout reflects the length of time in seconds that the LPS waits for response from the printer before displaying a timeout message. The default is 8 seconds, which may not be enough time if there are large graphics being sent to the printer that take longer than 8 seconds to load.

This property MUST be set in order to obtain the printer information (PrinterCount, PrinterName, PrinterPort, or PrinterAlias).

**Example**

```

Dim p as Integer
p = txtPrinterNum.Text
'Set the printer number, printer port and the printer timeout value
Call CWebX1.SetPrinter(p, txtPrinterPort, Text, txtPrinter.Text)

```

## Quantity Property

### Syntax

```

Object.Quantity
Type = short, read/write

```

### Description

The Quantity property is a read/write property that sets the amount of labels to print. The default for this property is one. The Quantity of labels is used to increment serial numbers and other incrementing

fields.

When using this property with a label that has an incrementing or decrementing serial number, it is important to note that changing this property DOES NOT create duplicate serial numbers. To create multiple copies of labels with identical serial numbers use the Duplicates Property.

#### Example

```
'This example ends up producing 10 labels
'with 2 duplicates of any incrementing or decrementing number
CWebX1.SetLabelName "Label1.lwl"
CWebX1.SetData 0, "ABC-123"
CWebX1.Quantity = 5
CWebX1.Duplicates = 2
CWebX1.PrintJob
```

## TrimLeadingSpaces Property

### Syntax

```
Object.TrimLeadingSpaces
Type = Boolean, read/write
```

### Description

The TrimLeadingSpaces property trims the leading spaces from data before it printed. If your program has control of the data, you should do any trimming or data manipulation before using the "SetData" Method. If you are going to use this property, it must be set before using the SetData Method. The default for this property is FALSE.

### Example

```
'This example ends up producing 10 labels
'The spaces before "ABC" are removed by the system
CWebX1.SetLabelName "Label1.lwl"
CWebX1.TrimLeadingSpaces=TRUE
CWebX1.SetData 0, " ABC-123"
CWebX1.PrintJob
```

## WebAddress Property

### Syntax

```
Object.WebAddress
Type = string
```

### Description

The web URL for the web server that connects to the LPS server. Where WebAddress is a character string containing the TCP/IP address in the standard format, or Hostname such as "www.loftwarelabeling.com".

**Example**

```
CWebX1.WebAddress = "172.15.0.99"
```

## WebPort Property

### Syntax

```
object.WebPort  
type = string
```

### Description

The WebPort Property sets the Web Port to be utilized in the transmission of the data. This is used if a Port other than the default (80) is used.

**Example**

```
CWebX1.WebPort = "8080"
```

## WebUserName Property

### Syntax

```
object.WebUserName
```

### Description

(OPTIONAL) Username for security (if required for access to the Web Server)

Default: null

**Example**

```
CWebX1.WebUserName = "UserOne"
```

## Internet ActiveX Methods

This section describes Loftware's ActiveX Methods.

## AppendJob Method

### Syntax

```
Object.AppendJob
```

### Description

AppendJob is used to queue up the current label that has been designated via the SetLabelName method calls, as well as Quantity, Duplicates, Pages, and/or PrinterNumber properties. AppendJob is used for batching label requests together for one server request, instead of many separate jobs.

**Example**

```
Private Sub cmdAppendJob_Click()
    Adds the current job to a queue
    CWebX1.AppendJob
End Sub
```

## ClearAllData Method

### Syntax

```
Object.ClearAllData
```

### Description

ClearAllData is used to clear all the data members for the current label, ensuring there are no 'sticky' data values being sent for the next job.

**Example**

```
Private Sub cmdCancel_Click()
    'Clears data queued
    CWebX1.ClearAllData
    'Removes all queued jobs
    CWebX1.ResetJob
End Sub
```

## Connect Method

### Syntax

```
Object.Connect
```

### Description

Connect the iX control to the Web Server and JSP. If the JSP is successful in establishing a session with LPS, Connect returns TRUE.

Otherwise, Connect returns FALSE and an ErrorEvent is generated.

**Note:** THIS METHOD SHOULD BE CALLED BEFORE ANY MANIPULATION OF THE CONTROL.

**Example**

```
Call CWebX1.Connect (txtURL.Text, txtPort.Text, txtUserName.Text, txtPassword.Text)
```

## Disconnect Method

### Syntax

```
Object.Disconnect
```

## Description

This method disconnects the iX Control from the Web and LPS connection. If there are no errors, Disconnect returns TRUE. If there is an error, Disconnect returns FALSE.

### Example

```
CWebX1.Disconnect
```

## PrintJob Method

### Syntax

```
Object.PrintJob
```

### Description

PrintJob is used to submit the current job to the LPS via the internet. iX client control creates a PAS file with all queued jobs and sends it to the LPS over the web. The current data is supplied via the SetData method calls, as well as Quantity, Duplicates, Pages, and/or PrinterNumber properties. The job is then created as a print stream by the LPS, sent back to the client control, and printed.

### Example

```
CWebX1.Quantity = txtQuantity.Text
CWebX1.Duplicates = txtDuplicates.Text
CWebX1.PrintJob
```

## ResetJob Method

### Syntax

```
Object.ResetJob
```

### Description

ResetJob is used to clear the entire job, including any labels that have been appended to the batch with the AppendJob method. This is useful for instances where the end user has changed their mind and wishes to cancel the job.

### Example

```
Private Sub cmdCancel_Click()
'Clears data queued
CWebX1.ClearAllData
'Removes all queued jobs
CWebX1.ResetJob
End Sub
```

## SetData Method

### Syntax

```
Object.SetData
```

## Description

The 'FieldCount', 'FieldName', 'FieldData', and 'FieldLength' properties describe the array that is created when the 'SetLabelName' method is invoked. SetData accesses this array and populates the field specified by either index number or field name with the data specified. Your program may not know ahead of time which and how many fields are in the label. This is why we allow you to iterate through the array with an index. If you do know your field names ahead of time, it is easier to set their data using the 'FieldName' property.

**Note:** You must do a 'SetLabelName' before you can set data using 'SetData'. You do not have to redo a 'SetLabelName' after doing a 'PrintJob', unless you are changing to a different label format.

### Example1

```
Call CWebX1.SetData(txtFieldName.Text, txtData.Text)
```

### Example 2

```
'setting data by iterating through field count instead of using field name
'this example sets every field to "sample data"
CWebX1.SetLabelName "ibm.LWL"
For I = 0 to CWebX1.FieldCount - 1
    CWebX1.SetData I, "sample data"
Next I
CWebX1.PrintJob
```

## SetLabelName Method

### Syntax

```
Object.SetLabelName
```

### Description

The SetLabelName method accesses the label across the web and retrieves the field information for the specified label format. This method also creates an array to hold the variable data for every variable field on the current label format. This array is accessed by index number or field name. If you do not know the field names ahead of time, you must use an index into the array. See 'FieldCount' and 'SetData' for more information on this.

**Note:** Because the array is contained within the control, it is Private. The only way to get information to and from the array is through the control's properties and methods. The Array that is created has the following structure:

FieldName() as a String

FieldLength() as a short

FieldData() as a String

The first index number of the array is zero.

### Example

```
CWebX1.SetLabelName "AIAG.LWL"
CWebX1.SetData "PARTNUMBER", "A100"
CWebX1.PrintJob
```

## SetPrinter Method

### Syntax

```
Object.SetPrinter
```

### Description

The SetPrinter Method sets the following parameters for the local printer:

- Index into local tables (the printer number)
- The local port (PrinterPort)
- A timeout value (PrinterTimeout)
- Sets the corresponding properties

#### Example

```
CWebX1.SetPrinter
Dim p As Integer
p = txtPrinterNum.Text
Call CWebX1.SetPrinter(p, txtPrinterPort.Text, txtPrinterTimeout.Text)
CWebX1.printerNumber = p
```

## Test Connection Method

### Syntax

```
Object.TestConnection
Type = Boolean
```

### Description

This method tests the URL/Port connection to servlet. Similar to the “Test” button on the WebClient Address/URL dialog box.

#### Example

```
Dim nPort as integer or short
Err.Clear
nPort = CInt(txtPort.Text)
Call CWebX1.TestConnection(txtURL.Text, nPort, txtUserName.Text, txtPassword.Text)
If (Err.Number <> 0) Then
    MsgBox ("Test Connection Failed!")
Else
    MsgBox ("Test Connection OK!")
End If
```

## TestPrinter Method

### Syntax

```
Object.TestPrinter
```



## Description

This method tests the local printer denoted by “PrinterNumber” by sending a formfeed to the printer. This is useful to see if your locally configured client-defined printer is configured / connected correctly. If communication is successful, a label ejects with the word TEST printed on the label, and the message “Check printer for the Label” is displayed. If there is no communication with the printer, a “Test Failed” message box is displayed.

### Example

```
Dim p As Integer
    p = txtPrinterNum.Text
CWebX1.TestPrinter (p)
```

## Internet ActiveX Events

This section describes Software's iX Events.

### AfterPrint Event

#### Syntax

```
Object_AfterPrint()
```

#### Description

The after print event is an event that is fired just after closing the port once the print stream has been sent, and is used for specialized printing needs.

### Example

```
CWebX1_AfterPrint
```

### BeforePrint Event

#### Syntax

```
Object_BeforePrint()
```

#### Description

The before print event is an event that is fired just before opening the port to send the print stream, and is used for specialized printing needs.

### Example

```
CWebX1_BeforePrint
```

### ErrorEvent Event

#### Syntax

```
Object_ErrorEvent()
```

## Description

Passes error messages back to the container during various operations. ErrorEvent items are critical and indicate that an error is generated (caught by error handling). An operation might send multiple error events, but the first occurrence of the ErrorEvent is the error that is passed back to the container to the error handling routines. See the ErrorEventIDs table for a breakdown of all warning events that are generated. Make sure that your program takes appropriate action after receiving one of these events. The safest bet when it comes to the error event is to end the program.

**Note:** It is critical that you trap the error event. This is especially true if the LPS server is running in a clustered environment and a failover occurs. Many of the ClientX methods throw critical errors during the failover transition. The following error trap handles this scenario:

### Example

```
'display a message box showing the error number and string
Private Sub CWebXl_ErrorEvent(ByVal errorID As Long, ByVal errorString As String)
    Debug.Print errorString
End Sub
```

## ErrorEvent Ids and Messages

(Also includes error returned from the control)

ID	Message	Explanation
25503	Critical Failure of job # jobNum	A critical failure of a job has been reported
25504	Error on printer prnternumber	The server is reporting a printer error on the listed printer. (Not currently used in the control)
25505	Printer Name: Not Found	The printer name was not located in the LPS printer configuration file
25506	Printer Port: Not Found	The printer port was not located in the LPS printer configuration file
25507	Printer ID: Not Found	The printer id was not located in the LPS printer configuration file .
25512	Failed to obtain the Computer Name!	The computer name is required to generate the unique jobname as well as unique filenames.
26500	Unable to open file.	Unable to open the label file
26501	Unable to open file 'file'.	Unable to open the label file
26502	Unable to change file mode for 'file'	Internal error when opening and parsing the label file. (Try again)

ID	Message	Explanation
26503	Tab marker information not located in file 'file'	The requested label file has been saved under a previous version. Open and save the label file to the newest version
26504	Label not set	An attempt to access a field or print a job with having called SetLabelName() first.
26505	Not a valid index number	An invalid index number. (<0 or greater than the count)
26506	Fieldname 'name' not found	A call to SetData or Fieldlength with the invalid fieldname 'name'.
26509	Not a Valid Printer Number	A negative printer number was passed
26510	Printer alias 'alias' not found	Printer alias was not found.
26514	Value cannot be negative.	Quantity, Duplicates, Pages, or PrinterNumber was set to a negative number.
26515	Could not determine the fieldname/position	Call to SetData with a zero-length string for a fieldname.
27501	Not connected to Server	Connect method must be called before calling other methods
27502	Error Retrieving Data from LPS	A communications level error occurred while transacting with the Print Server. Try the operation again, or disconnect
27503	Error processing request from LPS	An error occurred while processing a request on LPS. The textual message is included.
27504	Error Printer has not been locally configured	An attempt to print to a printer without calling SetPrint first.
27505	Error Control not properly initialized	An attempt to print a job without a valid connection.
27506	Error Print Job Totally Failed	The entire print request failed
27507	Failed to initialize the printer	Failed to connect to the printer, check configuration to ensure proper connection
27508	Failed to generate the print request	Internal error attempting to send the print request to LPS. Disconnect, reconnect and try again.
27509	Invalid UserName/Password	An attempt to connect to the Web Server failed with an invalid UserName and/or Password.

ID	Message	Explanation
27510	The connection with LPS has been dropped! Please restart.	The connection has been severed. LPS, Web Server or the Internet connection may have been shut down. Disconnect, reconnect and try again.
27511	Received error xx from Web Server/Servlet	An error from the Loftware Web Servlet has been reported. Please note the number, disconnect, reconnect, and then try again. If this persists, contact Loftware Technical Support for further assistance.

## InfoEvent Event

### Syntax

```
Object_InfoEvent()
```

### Description

The InfoEvent Event passes informational messages back to the container during various operations. InfoEvent items are used mainly as checkpoints and can be useful if displayed to the end user during initialization, etc. See the InfoEvent ID table for a breakdown of all info events that may be generated. This is a useful event during the development/debug process of your program. Once your program is fully debugged, you do not have to pay much attention to this event.

#### Example

```
Private Sub CWebXl_InfoEvent(ByVal infoID As Long, ByVal infoString As String)
    Debug.Print infoString
End Sub
```

### InfoEvent IDs and Messages

ID	Message	Explanation
25005	PasExt: ext	Displays the extension used for PAS files.
25006	Name: name	Displays the ServerName.
25007	Alias: alias	Displays the ServerAlias.
25008	Gathering Printer Information	Checkpoint before obtaining the printer info.
25009	Printer number	Displays the PrinterNumber
25010	Printer name:	Displays the PrinterName
25011	Printer Port: port	Displays the PrinterPort
25012	Printer ID: number	Displays the printer ID
25013	Printer Alias: alias	Displays the PrinterAlias

ID	Message	Explanation
25018	Getting information for entry: section	Lists the current section in the ini file being processed.
25019	Verified Path Exists	Path existence test OK.
25020	Verified Write Access	Path write test OK
25021	Verified Existence of type files	Files with extension of type were found in directory
25022	Verified Read Access for one file: file	Verified the ability to open a file for read (filename is displayed).
25023	Verified Read Access	Verified read access to a directory
25025	Verified File Exists	A file has been checked for existence
27001	Printed Job Number xx to printer on port	Job Printed Info Message

## OtherEvent Event

### Syntax

```
Object.OtherEvent()
```

### Description

Other events are not currently used, but they contain information that does not fit into the info, warning, or error categories.

#### Example

```
Private Sub CWebX1_OtherEvent(ByVal otherID As Long, ByVal otherString As String)
    Debug.Print otherString
End Sub
' Log these messages to an info multi-line textbox
txtOther.Text = otherString
End sub
```

## WarningEvent Event

### Syntax

```
Object.WarningEvent()
```

### Description

Passes warning messages back to the container during various operations. WarningEvent items are used to flag a missing item, etc., which is not detrimental to the operation, but is out of the ordinary. See the WarningEventID table for a breakdown of all warning events that may be generated.

#### Example

```
'display a message box showing the error number and string
Private Sub CWebX1_WarningEvent(ByVal warningID As Long, ByVal warningString As
```

```
String)
```

```
MsgBox("Warning # " & warningID & " has occurred. Message = " & warningString)
End sub
```

## WarningEvent IDs and Messages

ID	Message	Explanation
25250	Alias Not Found!	Server Alias was not located (optional entry) in the llmwcInt.ini file.
25254	PrinterAlias: Not Found	There was no Printer Alias defined for the particular printer.
25256	Could not locate any 'type' files	No files of type "type" were located when checking the path.
25257	Unable to test Read Access!	Since there were no files located, read access on files cannot be checked.
25258	Failed to obtain the Username of the currently logged in user	The username of the current logged in user is unable to be determined. (Used for job tracking, on the back end).
27251	There were errors processing Job!	There were errors encountered by LPS while processing the job/request.

## Internet ActiveX Reference Table

Name	Read	Write	Type	Default	Comment
AfterPrint Event	N/A	N/A	Event	N/A	Event fired just after closing port
BeforePrint Event	N/A	N/A	Event	N/A	Event fired just after opening port
AppendJob	N/A	N/A	Method	N/A	Batches label requests together.
CancelOperation	N/A	N/A	Property	N/A	Cancels data transmission
ClearAllData	N/A	N/A	Method	N/A	Clears all the data members for the current label
Connect	N/A	N/A	Method	N/A	Connects to the Web Server
Disconnect	N/A	N/A	Method	N/A	Disconnects from the Web Server
Duplicates	X	X	Property	1	Sets the amount of duplicate labels to print.
ErrorEvent	N/A	N/A	Event	N/A	Passes error messages back to the container.
Field Count	X		Property	N/A	Displays how many variable fields there are in the current label format

Name	Read	Write	Type	Default	Comment
FieldLength	X	N/A	Property	N/A	Displays the length of a specified field in the current label format
FieldName	X		Property	N/A	Displays the name of a specific field in the current label format.
InfoEvent	N/A	N/A	Event	N/A	Passes informational messages back to the container.
isConnected	N/A	N/A	Property	N/A	Checks to see if the program is connected to the Web Server
JobName	X	X	Property	Contains computer name	Reflects a unique identifier for the current job.
OtherEvent	N/A	N/A	Event	N/A	Contains information that does not fit into the info, warning or error categories
Pages	X	X	Property	1	Sets how many pages of labels are printed.
PrinterAlias	X		Property	N/A	The descriptive name for an assigned printer.
Printer Count	X		Property	N/A	The number of configured printers.
PrintJob	N/A	N/A	Method	N/A	Used to submit the current job to the LPS.
PrinterName	X		Property	N/A	Name of configured printer.
PrinterNumber	X	X	Property	1	Corresponds to configured Printers
PrinterPort	X		Property	N/A	List of configured printer ports
PrinterTimeout	X		Property	N/A	Sets the time in seconds before Printer Timeout message is displayed.
Quantity	X	X	Property	1	Sets the amount of labels to print.
ResetJob	N/A	N/A	Method	N/A	Clears the entire job, including appended labels.
SetData	N/A	N/A	Method	N/A	Populates the field by index number or field name.
SetLabelName	N/A	N/A	Method	N/A	Retrieves field information for the specified label format.
SetPrinter	N/A	N/A	Method	N/A	Tells the control which printer to use.
TestConnection	N/A		Property	0	Tests the connection to the Web Server

<b>Name</b>	<b>Read</b>	<b>Write</b>	<b>Type</b>	<b>Default</b>	<b>Comment</b>
TestPrinter	N/A	N/A	Method		Sends a test to the printer to see if properly configured
TrimLeading Spaces			Property		Trims spaces off of label data
WarningEvent	N/A	N/A	Event	N/A	Passes warning messages back to the container during various operations.
WebAddress	X		Property		The Web IP Address or URL for the web server that connects to the LPS server.
WebPort	X		Property	80	Sets the WebPort number if you are not using the default
WebUserName			Property	Null	Username for security (Used if required for access to the Server)



This page intentionally left blank

Using Software's .NET Control, you can integrate Software's barcodes printing modules directly into your own applications.

These controls diminish the level of knowledge and expertise required to connect and print to stand-alone and networked barcodes label printers. Development languages such as C#, J#, managed C++, and VB.NET can utilize this technology as long as they are a CLR compliant language.

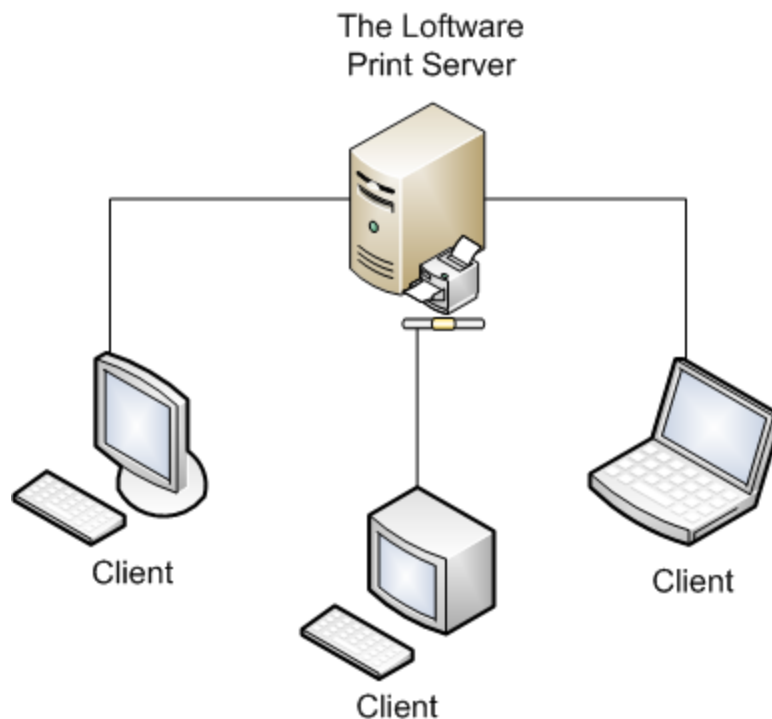


Figure 11.1: Computers using one of the Client Controls to offload print jobs to the LPS

This section documents the Software .NET Control. Like the other Software Client Controls, this control has a thin footprint because it does not require that the Software Print Server be installed on the same computer as the control. It is called a client control because it acts as a client to the Software Print Server. You can use this control when your application is running in several places, needs to access many printers, and requires a small footprint.

The Software .NET Control utilizes the Microsoft .NET© framework. This control is designed for use in .NET applications where ActiveX Client Controls may not be suitable.

## Considerations when using the .NET Control

The Software .NET Control acts similar to the way other Software Client Controls do. However, for this control, the file drop functionality is not available. A client connection from the control is easier to control, and more scalable. Considerations in the use of the .NET Control include:

- The LPS is logged in to as a client application.
- The .NET Control uses the \*.XML file type for print requests.
- Print requests are created programmatically by providing data for each field on the selected label.
- Quantity, duplicates, pages, and field data are set for each label.
- The print job or loaded label only exists in memory while the client is logged in. All of the print jobs are deleted from memory and the label is unloaded if logged out.
- One label at a time may be loaded.

## Installation and Use of the Software .NET Control

### Pre-Installation Requirements

- Microsoft Visual Studio .NET 2013 is required to open and build any of the Software .NET Control sample solutions listed below.
- Microsoft .NET Framework 4.5.1 is required for development using the Software .NET Control.

**Note:** If you do not already have the Microsoft .NET Framework 4.5.1 installed, it is installed during the Software .NET Control install.

- One of the supported Operating Systems.
- The Software Print Server Premier Edition, with available client seats.

#### See Also

System Requirements in the *Software Print Server and Label Manager Installation Guide*.

### Sample Programs and Read Me Files

Before attempting to use the Software .NET Control, please review the sample programs and their associated Read Me files. By default, the sample programs can be found at:

```
\Software Labeling\Software .NET Control\Samples
```

There are three sample programs in this folder:

#### C# Samples

- TestClientSimple - This project is an implementation of Software's .NET Control written in C#. Its purpose is to show you how to use the majority of the methods that Software's .NET Control

offers.

- **TestClientAdvanced** - The TestClientCSharp project is an implementation of the .NET Control written in C#. It is based on Loftware's On Demand Print Client and the Web Client.

**Important:** You may need to make some changes for the sample applications to work properly:

Change the labels folder to C:\Program Files (x86)\Loftware Labeling

Disable IP Version 6 (IPv6). For more information, see Microsoft under "[External Links](#)" on page 283.

If the samples are running on the same computer as the LPS, you will need to disable IPv6 on the LAN.

#### **Unbind IPv6 from a specific network adapter**

You can unbind IPv6 from a specific network adapter in Local Area Connection Properties. To do this, follow these steps:

1. Click **Start**, and then click **Control Panel**.
2. Click **Network and Sharing Center**.
3. In the View your active networks area, click **Local Area Connection**, and then click **Properties**.
4. On the **Networking** tab, clear the **Internet Protocol Version 6 (TCP/IPv6)** check box, and then click **OK**.

## **VB .NET Samples**

- **Loftware .NET Control VB Test Client** - The VBTestClient project is an implementation of Loftware's .NET Control written in VB.NET. Its purpose is to show you how to use the majority of the methods that Loftware's .NET Control offers.

Review the sample program's Read Me files to help you understand build steps and special notes the Loftware .NET Control.

## **Installing the Loftware .NET Control**

The first part of the install checks the computer for Microsoft Windows® Installer. If the correct version is not found on the computer, the Loftware .NET Control install updates the Windows installer version. You may have to reboot your computer.

The second part of the install checks the system for the existence of the Microsoft® .NET framework. If the correct version does not exist, the .Net framework is installed on the server. A series of screen prompts are displayed. The installation of the .NET framework may take up to 10 minutes depending on your computer's hardware and operating system. You may have to reboot your computer after the .NET framework installation.

The third part of the install installs the Loftware .NET Control application and samples to:

Loftware Labeling\Loftware .NET Control

## Installing the .NET Control

1. Open the Software .NET Control folder from the CD or download package, and run Setup.
2. Click **Next** at the **Welcome** page.
3. Choose **Complete** or **Custom** install, and click **Next** from the **Setup Type** page.
4. Click **Install** at the **Ready to Install** page.
5. Click **Finish** at the **Install Shield Wizard Completed** page.

## Configuring the Software Print Server for use with the .NET Control

1. Configure all connected printers using Software Label Manager.
2. Design the necessary label formats using Software Label Manager, and make the labels available to the Software Print Server.  
The Options | File Locations menu in Software Label Manager allows you to set network locations for your label files. For example, the shared Software Labeling\Labels folder on the Software Print Server server.
3. Verify that the LPS is working by building a simple label and manually creating a .pas file for it. Copy your pas file to the directory that LPS is scanning and your label should print.

### Related Information

For information on configuring printers, refer to *Device Connections* in the *Software Label Manager User's Guide*.  
For troubleshooting information, refer to *The Software Print Server* section of this guide.

## Using the Software .NET Control

1. Install the Software .NET Control on a client computer.
2. Load one of the sample programs. The sample programs for the Software .NET Control are found by default under:

\Software Labeling\Software .NET Control\Samples

3. Create a new project using Microsoft Visual Studio.
4. Add the Software .NET Control to the new project. For example, follow these steps if you are using Microsoft Visual Studio:

### Add a reference to the Software.LLMControl.dll.

1. Right-click **References** in Solution Explorer and select **Add Reference**.
2. Select the **.NET** tab in the **Add Reference** dialog, click **Browse** and browse to ..\Software Labeling\Software .NET Control.
3. Select the Software.LLMControl.dll, click **Open**, and then click **OK** in the **Add Reference** dialog. A reference to Software.LLMControl is now shown under the References in your project.

## If your project is a Windows Application, follow these steps:

1. Open the main form of your project in design view.
2. Right-click and select **Add/Remove Items**.
3. Select the **.NET Framework Components** tab in the **Customize Toolbox** dialog, click **Browse** and browse to `..\Software Labeling\Software .NET Control`.
4. Select the `Software.LLMControl.dll` and click **Open**. `LoftClient` is then added to the list of .NET Framework Components.
5. Verify that **LoftClient** is selected, and click **OK**.
6. Select the **LoftClient** component under **My User Controls** and place it on the main form of your project. An item named **LoftClient1** is displayed in a window below your form.
7. Right-click **LoftClient1** and select **Properties** if you want to change its default name.

## If your project is a Console Application

- During the initialization of your application, create an instance of the Software .NET Control. For example:

```
Software.LLMControl.LoftClient m_LoftClient = new Software.LLMControl.LoftClient();
```

**Important:** The `Software.LLMControl.DLL` must be in the same directory as the executable for your project. Copy and paste the DLL to your debug, release, or bin folders as necessary.

## Distributing the Software .NET Control

The Software .NET Control can be distributed with your application(s) to anyone who requires it. When building an install program for your application, include the required `Software.LLMControl.dll` with it. Your setup program should take care of installing the `Software.LLMControl.dll`, thereby avoiding having to run two setup programs.

## Using the .NET Control in a Web Application

It is possible to use the Software .NET Control in a web application such as ASP.NET. To do so the project must contain a `Web.config` file. To add a `Web.config` file to a Microsoft Visual Studio project follow these steps:

1. Select the **Project** menu.
2. Select **Add New Item**.
3. In the **Add New Item** dialog, select **Workflow**, and select **Web Configuration File**.
4. Click **Open**.

The `Web.config` file is an XML file. To use the .NET Control in a web application it is necessary to edit this file. Open it and add the following to the `<configuration>` section:

```
<appSettings>  
<add key="assemblyPath" value="C:\TEMP\dotNetControl"/>  
</appSettings>
```

The assemblyPath value can be set to any directory but the directory must contain the Software.LLMControl.dll.

The DLL can reside anywhere\* as long as the assemblyPath value points to its location. The Software.LLMControl.dll must reside in the same directory as the web application.

**\*Note:** Even if the Software.LLMControl.dll is in the same directory as the web application, the Web.Config file still needs to exist with the assemblyPath value pointing to the directory of the web application.

## Software .NET Control Methods

Methods are used when you want to get information, format information, or send information. If you see a method definition with two parameter sets, that method is overloaded. The methods described in this section are used in the Software .NET Control.

### AppendJob

#### Declaration

```
Public Sub AppendJob()
```

#### Syntax

```
control.AppendJob()
```

#### Description

AppendJob takes the current job in memory and saves it on a list, allowing you to work on a new job and still be able to print the other jobs.

**Note:** The .NET control supports only one printer per print job. Appended jobs must be sent to the same printer as the first specified. All requests will be sent to the first printer specified.

#### Example

```
Private Sub Btnappend_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles btnappend.Click  
LoftClient.PrinterName = "dotnetprinter"  
LoftClient.JobName = "job2723"  
LoftClient.Quantity = 1  
LoftClient.Duplicates = 1  
LoftClient.Pages = 1  
LoftClient.SetData("text0000", TextBox1.Text) 'text0000 is the name  
of the field  
LoftClient.SetData(1, TextBox2.Text) '1 is the index number of the  
field  
LoftClient.AppendJob()'Appends this job to the queue  
End Sub
```

## ClearData

### Declaration

```
Public Sub ClearData()
```

### Syntax

```
control.ClearData()
```

### Description

The ClearData method clears all of the user-entered data from the current job.

#### Example

```
Private Sub Btnappend_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnappend.Click
LoftClient.PrinterName = "dotnetprinter"
LoftClient.JobName = "job2723"
LoftClient.Quantity = 1
LoftClient.Duplicates = 1
LoftClient.Pages = 1
LoftClient.SetData("text0000", TextBox1.Text) 'text0000 = name of the field
LoftClient.SetData(1, TextBox2.Text) ' 1 is the index number of the field
LoftClient.ClearData() 'Clears the data that was entered for current job
End Sub
```

## GetFieldIndex

### Declaration

```
Public Function GetFieldIndex(ByVal strFieldName As String) As Integer
```

### Syntax

```
nFieldIndex = control.GetFieldIndex(strFieldName)
```

### Description

The GetFieldIndex method allows you to obtain the index of the field based on the field name.



**Example**

```

Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Handles MyBase.Load
    Dim iCount As Integer
    Dim lblFieldTest As String
    LoftClient.Login("172.16.0.93", 2723)
    ListLabels("")
    printgrid()
    LoftClient.GetLabel("c:\Program files\Software Labeling\Labels\dotnet.lwl")
    LoftClient.LabelDtd = ("c:\Program files\Software
        Labeling\batch\label.dtd")
    lblgrid()
    lblindex0.Text = LoftClient.GetFieldIndex("Text0000").ToString()
    lblindex1.Text = LoftClient.GetFieldIndex("Text0001").ToString()
    lblserver.Text = LoftClient.CurrentServer
End Sub

```

## GetFieldMaxLength

### Declarations

```

Public Function GetFieldMaxLength(ByVal iFieldNum As Integer) As Integer
Public Function GetFieldMaxLength(ByVal strFieldName As String) As Integer

```

### Syntax

```
nMaxFieldLen = control.GetMaxFieldLength(strFieldName)
```

or

```
nMaxFieldLen = control.GetMaxFieldLength(nFieldIndex)
```

### Description

The GetFieldMaxLength method takes the Index or Name of the fields and returns the maximum number of allowed characters for that field.

**Example**

```

' gets user entered data from the list view and puts it into an edit box
' restricts the amount of data a user can type into a edit box
Private Sub lstLabelData_SelectedIndexChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles lstLabelData.SelectedIndexChanged
    Dim iMaxFieldLen As Integer
    'get the selected item's text, put it into the data entry edit box
    If lstLabelData.SelectedIndices.Count Then
        ' get the field max length so we can constrain input
        iMaxFieldLen=mLlmClient1.GetFieldMaxLength(lstLabelData.SelectedIndices.
            Item(0))
        DataEntry.MaxLength = iMaxFieldLen
        If lstLabelData.SelectedItems(0).Text.Length() Then
            DataEntry.Text = lstLabelData.SelectedItems(0).Text
        End If
    End If
End Sub

```

## GetFieldName

### Declaration

```
Public Function GetFieldName(ByVal iFieldNum As Integer) As String
```

### Syntax

```
strFieldName = control.GetFieldName(nFieldIndex)
```

### Description

The GetFieldName method takes the Index of the field and returns that Name. This function allows you to quickly iterate through all the fields after loading a label using the FieldCount property.

#### Example

```
' get the label information and add it to the list view
Private Sub GetFields()
    Dim i As Integer
    Dim Item As ListViewItem
    Dim sFieldName As String
    Dim iMaxFieldLen As Integer
    ' get the label whose name is stored in mLabelName
    mLlmClient1.GetLabel(mLabelName)
    For i = 0 To mLlmClient1.FieldCount - 1
        Item = New ListViewItem("")
        ' get the field name from the control
        sFieldName = mLlmClient1.GetFieldName(i)
        ' get the max field length from the control
        iMaxFieldLen = mLlmClient1.GetFieldMaxLength(i)
        ' add the field name and max field length to the list view
        Item.SubItems.Add(sFieldName)
        Item.SubItems.Add(System.Convert.ToString(iMaxFieldLen))
        lstLabelData.Items.Add(Item)
    Next
    ' resize the list view columns to fit
    lstLabelData.Columns(0).Width = lstLabelData.Width / 3
    lstLabelData.Columns(1).Width = lstLabelData.Width / 3
    lstLabelData.Columns(2).Width = lstLabelData.Width / 3
End Sub
```

## GetLabel

### Declaration

```
Public Function GetLabel(ByVal strLabelName As String) As Boolean
```

### Syntax

```
bResult = control.GetLabel(strLabelName)
```

## Description

The GetLabel method downloads the label information from the LPS and sets it as the currently loaded label in memory. This method must be called before you begin work on a job.

### Example

```
' get the label information and add it to the list view
Private Sub GetFields()
    Dim i As Integer
    Dim Item As ListViewItem
    Dim sFieldName As String
    Dim iMaxFieldLen As Integer
    ' get the label whose name is stored in mLabelName
    mLlmClient1.GetLabel(mLabelName)
    For i = 0 To mLlmClient1.FieldCount - 1
        Item = New ListViewItem("")
        ' get the field name from the control
        sFieldName = mLlmClient1.GetFieldName(i)
        ' get the max field length from the control
        iMaxFieldLen = mLlmClient1.GetFieldMaxLength(i)
        ' add the field name and max field length to the list view
        Item.SubItems.Add(sFieldName)
        Item.SubItems.Add(System.Convert.ToString(iMaxFieldLen))
        lstLabelData.Items.Add(Item)
    Next
    ' resize the list view columns to fit
    lstLabelData.Columns(0).Width = lstLabelData.Width / 3
    lstLabelData.Columns(1).Width = lstLabelData.Width / 3
    lstLabelData.Columns(2).Width = lstLabelData.Width / 3
End Sub
```

## GetLabelList

### Declaration

```
Public Function GetLabelList(ByVal strSubDir As String, ByRef strLabelList() As
String) As Boolean
```

### Syntax

```
bResult = control.GetLabelList(strFolder, arrLabelNames)
```

### Description

The GetLabelList method returns a list of all available labels on the LPS as a string array. You may pass in a path if you wish to specify another path on the LPS for the labels directory. Not specifying a path defaults it to the current label directory.

**Example**

```
' gets a list of available labels and displays them in a listbox
Private Sub GetLabels()
    Dim LabelList() As String
    Dim Folder As String
    Dim ErrorBox As MessageBox
    '
    lstLabels.DataSource = Nothing
    ' get a list of available labels
    If LlmClient1.GetLabelList(Folder, LabelList) <> True Then
        ErrorBox.Show("Failed to get list of labels from server")
        Return
    End If
    lstLabels.DataSource = LabelList
End Sub
```

## GetLabelListExt

### Declaration

```
Public Function GetLabelListExt(ByVal strSubDir As String, ByRef strLabelList() As String) As Boolean
```

### Syntax

```
bResult = control.GetLabelListRxt(strFolder, arrLabelNames)
```

### Description

The GetLabelListExt method is similar to the GetLabelList method, except that GetLabelListExt returns a list with file extensions on the LPS as a string array. You may pass in a path if you wish to specify another path on the LPS for the labels directory. No path defaults to the current label directory.

**Example**

```
' gets a list of available labels and displays them in a listbox
Private Sub GetLabelsAndTags()
    Dim LabelList() As String
    Dim Folder As String
    Dim ErrorBox As MessageBox
    lstLabels.DataSource = Nothing
    ' get a list of available labels with file extensions
    If LlmClient1.GetLabelListExt(Folder, LabelList) <> True Then
        ErrorBox.Show("Failed to get list of labels from server")
        Return
    End If
    lstLabels.DataSource = LabelList
End Sub
```

## GetLabelsPrinterID

### Declaration

```
Public Function GetLabelsPrinterID(ByVal strLabelName As String, ByRef strPrintID As String) As Boolean
```

### Syntax

```
bResult = control.GetLabelsPrinterID(strLabelName, strPrinterID)
```

### Description

GetLabelsPrinterID passes in a label name (with path, optional) and the ID of the printer the label was designed for comes back as strPrinterID.

#### Example

```
' get the printer ID the label was designed for
Private Sub GetLabelsPrinter(strLabelName) As String
    Dim strID As String
    ' get the printer ID
    If LlmClient1.GetLabelsPrinterID(strLabelName, strID) <> True Then
        ErrorBox.Show("Failed to get printer ID")
        Return
    Else
        Return(strID)
    End Sub
```

## GetPrinterByNumber

### Declaration

```
Public Function GetPrinterByNumber(ByVal iPrinterNumber As Integer, ByRef strPrintName As String, ByRef strPrintAlias As String, ByRef strPrintPort As String) As Boolean

Public Function GetPrinterByNumber(ByVal iPrinterNumber As Integer, ByRef strPrintName As String, ByRef strPrintAlias As String, ByRef strPrintPort As String, ByRef strPrintID As String) As Boolean
```

### Syntax

```
bResult = control.GetPrinterByNumber(nPrinterNumber, strPrinterName, strPrinterAlias, strPrinterPort)

bResult = control.GetPrinterByNumber(nPrinterNumber, strPrinterName, strPrinterAlias, strPrinterPort, strPrinterID)
```

### Description

The first (original) GetPrinterByNumber method described previously takes in the LLM Printer number and gives back the name, alias (if any) and port of the specified printer as strings.

The second `GetPrinterByNumber` method is a method override and takes in the LLM Printer number and gives back the name, alias (if any), port, and printer ID of the specified printer as strings.

#### Example of the first `GetPrinterByNumber` method

```
' gets a list of available printers and displays them in a list view
Private Sub GetPrinters()
    Dim sPrinter As String
    Dim sAlias As String
    Dim sPort As String
    Dim i As Integer
    Dim ErrorBox As MessageBox
    Dim Item As ListViewItem
    lstPrinters.Items.Clear()
    ' get a list of available printers
    If LlmClient1.GetPrinters(mPrinterNumbers) <> True Then
        ErrorBox.Show("Failed to get printer list")
        Return
    End If
    For i = 0 To mPrinterNumbers.Length - 1
        ' get the printer name, alias and port
        If LlmClient1.GetPrinterByNumber(mPrinterNumbers(i), sPrinter, sAlias,
            sPort) <> True Then
            ErrorBox.Show("Failed to get printer name")
            Return
        Else
            Item = New ListViewItem(" " + System.Convert.ToString(mPrinterNumbers(i)) + " ")
            Item.SubItems.Add(sPrinter)
            Item.SubItems.Add(sAlias)
            Item.SubItems.Add(sPort)
            lstPrinters.Items.Add(Item)
        End If
    Next
    ' set the width of the listview columns
    lstPrinters.Columns(0).Width = lstPrinters.Width / 4
    lstPrinters.Columns(1).Width = lstPrinters.Width / 4
    lstPrinters.Columns(2).Width = lstPrinters.Width / 4
    lstPrinters.Columns(3).Width = lstPrinters.Width / 4
End Sub
```

**Example of the GetPrinterByNumber override**

```
' gets a list of available printers and displays them in a list view
Private Sub GetPrinters()
    Dim sPrinter As String
    Dim sAlias As String
    Dim sPort As String
    Dim sPrinterID As String
    Dim i As Integer
    Dim ErrorBox As MessageBox
    Dim Item As ListViewItem
    lstPrinters.Items.Clear()
    ' get a list of available printers
    If LlmClient1.GetPrinters(mPrinterNumbers) <> True Then
        ErrorBox.Show("Failed to get printer list")
        Return
    End If
    For i = 0 To mPrinterNumbers.Length - 1
        ' get the printer name, alias, port, printer ID
        If LlmClient1.GetPrinterByNumber(mPrinterNumbers(i), sPrinter, sAlias,
            sPort, sPrinterID) <> True Then
            ErrorBox.Show("Failed to get printer name")
            Return
        Else
            Item = New ListViewItem(" " + System.Convert.ToString(mPrinterNumbers(i))
                + " ")
            Item.SubItems.Add(sPrinter)
            Item.SubItems.Add(sAlias)
            Item.SubItems.Add(sPort)
            Item.SubItems.Add(sPrinterID)
            lstPrinters.Items.Add(Item)
        End If
    Next
    ' set the width of the listview columns
    lstPrinters.Columns(0).Width = lstPrinters.Width / 4
    lstPrinters.Columns(1).Width = lstPrinters.Width / 4
    lstPrinters.Columns(2).Width = lstPrinters.Width / 4
    lstPrinters.Columns(3).Width = lstPrinters.Width / 4
End Sub
```

## GetPrinters

### Declaration

```
Public Function GetPrinters(ByRef iPrinterList() As Integer) As Boolean
```

### Syntax

```
bResult = control.GetPrinters(arrPrinterNumbers)
```

### Description

The GetPrinters method obtains a list of all configured printers from the LPS and returns a list of the configured printer numbers as an int. This int should be used to obtain the printer information using the

GetPrinterByNumber function call.

### Example

```
' gets a list of available printers and displays them in a listview
Private Sub GetPrinters()
    Dim sPrinter As String
    Dim sAlias As String
    Dim sPort As String
    Dim i As Integer
    Dim ErrorBox As MessageBox
    Dim Item As ListViewItem
    lstPrinters.Items.Clear()
    ' get a list of available printers
    If LlmClient1.GetPrinters(mPrinterNumbers) <> True Then
        ErrorBox.Show("Failed to get printer list")
        Return
    End If
    For i = 0 To mPrinterNumbers.Length - 1
        ' get the printer name, alias and port
        If LlmClient1.GetPrinterByNumber(mPrinterNumbers(i), sPrinter, sAlias,
            sPort) <> True Then
            ErrorBox.Show("Failed to get printer name")
            Return
        Else
            Item = New ListViewItem(" " +
                System.Convert.ToString(mPrinterNumbers(i)) + " ")
            Item.SubItems.Add(sPrinter)
            Item.SubItems.Add(sAlias)
            Item.SubItems.Add(sPort)
            lstPrinters.Items.Add(Item)
        End If
    Next
    ' set the width of the listview columns
    lstPrinters.Columns(0).Width = lstPrinters.Width / 4
    lstPrinters.Columns(1).Width = lstPrinters.Width / 4
    lstPrinters.Columns(2).Width = lstPrinters.Width / 4
    lstPrinters.Columns(3).Width = lstPrinters.Width / 4
End Sub
```

## KillJobs

### Declaration

```
Public Sub KillJobs(ByVal bUnloadLabel As Boolean)
```

### Syntax

```
control.KillJobs(True)
```

### Description

The KillJobs method removes all the jobs from memory (saved via AppendJob) and clears the job you are currently working on. If you pass in true to the parameter, then KillJobs also “unloads” the currently



loaded label from memory.

#### Example

```
Private Sub btnkillJobs_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnkillJobs.Click
    LoftClient.KillJobs(False) 'clears jobs from the queue if set to true it
        will also unload the label from memory
End Sub
```

## Login

### Declaration

```
Public Function Login(ByVal strAddr As String, ByVal iPort As Integer) As Boolean
```

### Syntax

```
bResult = control.Login(strIpAddress, nPort)
```

### Description

The Login method takes the server name or IP address (string) and port number (int) and attempts to log in as a client to the LPS.

#### Example

```
' logs the user in or out in response to clicking the login button
Private Sub btnLogin_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnLogin.Click
    ' if we're logged in, log us out
    If LlmClient1.LoggedIn() Then
        btnLogin.Text = "Login"
        LlmClient1.Logout()
        lstPrinters.Items.Clear()
        lstLabels.DataSource = Nothing
    Else ' otherwise, log us in
        btnLogin.Text = "Logout"
        LlmClient1.Login(IPAddress.Text, 2723)
        GetLabels()
        GetPrinters()
    End If
End Sub
```

## Logout

### Declaration

```
Public Sub Logout()
```

### Syntax

```
control.Logout()
```

## Description

The Logout method sends a logout request to the LPS and then closes out the connected socket.

### Example

```
'Sends a logout request to the LPS and then closes out the connected socket.
Private Sub btnlogout_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnlogout.Click
    LoftClient1.Logout()
End Sub
```

## PrintJob

### Declaration

```
Public Function PrintJob(ByRef prtJobRsp As Loftware.LLMControl.PrintJobResponse) As
Integer
```

### Syntax

```
nJobNumber = LoftClient1.PrintJob(prtJobRsp)
```

or, if you don't want data back:

```
nJobNumber = LoftClient1.PrintJob()
```

## Description

The PrintJob method prints all of the jobs in memory, including the one currently being worked on. When PrintJob returns, the optional prtJobRsp object contains data that can be used to determine the result of the submitted job. See Class PrintJobResponse for more information.

**Note:** The .NET control supports only one printer per print job. Appended jobs must be sent to the same printer as the first specified. All requests will be sent to the first printer specified.

### Example

```
Private Sub btnPrint_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BtnPrint.Click
    Dim prtJobRsp As Loftware.LLMControl.PrintJobResponse
    Dim nJobNum As Integer
    LoftClient1.GetLabel("TestLabel.lwl")
    LoftClient1.SetData("TestField", "TestData")
    LoftClient1.PrinterNumber = 1
    nJobNum = LoftClient1.PrintJob(prtJobRsp)
    'nJobNum will be 0 if it's a 'critical error'
    'therefore, if it is not 0, check the StatusCode before trying
    'to grab data
    if ((nJobNum <> 0) And (prtJobRsp.StatusCode = 4))
        m_strXmlData = prtJobRsp.XmlData
    'now parse whatever data you want out of the XML
    Else
        MessageBox("Errors Encountered", MsgBoxStyle.Critical)
    End If
End Sub
```

## Class PrintJobResponse

### Declaration

```
Public Class PrintJobResponse
```

### Syntax

```
Dim prtJobRsp As Software.LLMControl.PrintJobResponse
```

### Description

The PrintJobResponse class is passed into the PrintJob method to obtain valuable data from the LPS in response to a print job request. At the time of this writing, the only data that is passed back is the EPC data related to RFID printing, and pass/fail job status. Future versions will have the ability to pass back other kinds of data. This is a synchronous call and may be blocked if the LPS is under a very heavy load. In other words, your code will not continue until the LPS has printed the request. For this reason, we suggest placing your print job code in a separate thread so that your program can be doing other things in the event it has to wait.

The PrintJobResponse class contains the following properties:

Property	Description
public int StatusCode	Each print job request results in one of the following status codes: 2 = Critical Failure 4 = Printed 6 = Printed with Errors 8 = Printer Error
public string StatusMessage	Each print job request results in one of the following status messages: "Critical Failure" "Printed" "Printed with Errors" "Printer Error"
public string ErrorMessage	Contains the error message if the print job request encountered any errors; otherwise, it is empty.
public int PrinterNumber	Printer number of the print job request
public int JobNumber	Job number of the print job request assigned by the LPS
public object Reserved	Empty, reserved for future use
public string XmlData	Response to the print job request from the LPS in XML form. May contain EPC data, see the following examples of possible XmlData values.

**Note:** To properly parse the XML response data, the LoftDataResp.xsd file must be copied to the path containing the executable file of the application that uses the .NET Control. If the XML responses are saved to disk, the LoftDataResp.xsd file must be copied to the same directory the XML responses are saved to. The LoftDataResp.xsd file is installed to the "Loftware Labeling\Batch" directory.

#### Example 1 – Single request containing EPC data

```
<?xml version="1.0" encoding="utf-8" ?>
<job xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="LoftDataResp.xsd" name="20041119153416000011709_
RKNOWLES-XP2" jobnumber="11709" device="6">
<status code="4" text="" />
<request number="1" LabelNumber="1">
<field name="RFID" value="B80F600390000064">
<property name="EncodingType" value="SGTIN-64"/>
<property name="FilterValue" value="7"/>
<property name="CompanyPrefixIndex" value="123"/>
<property name="ItemReference" value="456"/>
<property name="SerialNumber" value="100"/>
</field>
</request>
</job>
```

**Example 2 – Stacked request containing EPC data**

```
<?xml version="1.0" encoding="utf-8" ?>
<job xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="LoftDataResp.xsd" name="20041119153416000011709_
RKNOWLES-XP2" jobnumber="11709" device="6">
<status code="4" text="" />
<request number="1" LabelNumber="1">
<field name="RFID" value="B80F600390000064">
<property name="EncodingType" value="SGTIN-64"/>
<property name="FilterValue" value="7"/>
<property name="CompanyPrefixIndex" value="123"/>
<property name="ItemReference" value="456"/>
<property name="SerialNumber" value="100"/>
</field>
</request>
<request number="1" LabelNumber="2">
<field name="RFID" value="B80F600390000065">
<property name="EncodingType" value="SGTIN-64"/>
<property name="FilterValue" value="7"/>
<property name="CompanyPrefixIndex" value="123"/>
<property name="ItemReference" value="456"/>
<property name="SerialNumber" value="101"/>
</field>
</request>
<request number="1" LabelNumber="3">
<field name="RFID" value="B80F600390000066">
<property name="EncodingType" value="SGTIN-64"/>
<property name="FilterValue" value="7"/>
<property name="CompanyPrefixIndex" value="123"/>
<property name="ItemReference" value="456"/>
<property name="SerialNumber" value="102"/>
</field>
</request>
</job>
```

**Example 3 – Single request without EPC data**

```
<?xml version="1.0" encoding="utf-8" ?>
<job xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="LoftDataResp.xsd" name="20041201100415000011798_
RKNOWLES-XP2" jobnumber="11798" device="1">
<status code="2" text="PAS File Error" />
</job>
```

## SetData

### Declaration

```
Public Sub SetData(ByVal iFieldToSet As Integer, ByVal strData As String)
```

### Syntax

```
control.SetData(nFieldIndex, strDataValue)
```

or

```
control.SetData(strFieldName, strDataValue)
```

## Description

The SetData method takes the name or index of the field the user wishes to set the data for as well as the data as a string.

### Example

```
Private Sub btnprint_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Btnprint.Click
    LoftClient.GetLabel("c:\Program files\Loftware Labeling\Labels\dotnet.lwl")
    LoftClient.PrinterNumber = 1
    LoftClient.JobName = "job2723"
    LoftClient.Quantity = 1
    LoftClient.Duplicates = 1
    LoftClient.Pages = 1
    LoftClient.SetData("text0000", TextBox1.Text) 'text0000 is the name of the
        field
    LoftClient.SetData(1, TextBox2.Text) ' 1 is the index number of the field
    LoftClient.PrintJob()
End Sub
```

## Loftware .NET Control Properties

This section describes Loftware .NET Control properties and their access abilities (get, set, or both). The Loftware .NET Control properties refer to the currently loaded label.

### CurrentLabel

#### Declaration

```
Public ReadOnly Property CurrentLabel() As String
```

#### Syntax

```
strCurLabel = control.CurrentLabel
```

#### Description

The CurrentLabel property maps directly to the currently loaded label. This is only valid after calling GetLabel().

#### Example

```
Private Sub lblgrid() '(ByVal sender As System.Object, ByVal e As System)
    lbllabel.Text = LoftClient.CurrentLabel
End Sub
```

### CurrentServer

#### Declaration

```
Public ReadOnly Property CurrentServer() As String
```

#### Syntax

```
strServer = control.CurrentServer
```

#### Description

The CurrentServer property is the name or IP address of the LPS to which you are currently connected.

#### Example

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Handles MyBase.Load
    LoftClient.Login("172.16.0.93", 2723)
    ListLabels("")
    printgrid()
    LoftClient.GetLabel("c:\Program files\Loftware
        Labeling\Labels\dotnet.lwl")
    lblgrid()
    lblserver.Text = LoftClient.CurrentServer
End Sub
```

## Duplicates

### Declaration

```
Public Property Duplicates() As Integer
```

### Syntax

```
control.Duplicates=1
```

### Description

The Duplicates property allows you to both get and set the number of duplicates you want for the current job. This is only valid after calling GetLabel().

#### Example

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Handles MyBase.Load
        LoftClient.Duplicates=1
    End Sub
```

## FieldCount

### Declaration

```
Public ReadOnly Property FieldCount() As Integer
```

### Syntax

```
nFieldCount = control.FieldCount
```

### Description

The FieldCount property allows you to get the total number of fields in the currently loaded label. This is only valid after calling GetLabel().



**Example**

```
' get the label information and add it to the list view
Private Sub GetFields()
    Dim i As Integer
    Dim Item As ListViewItem
    Dim sFieldName As String
    Dim iMaxFieldLen As Integer
    ' get the label whose name is stored in mLabelName
    mLlmClient1.GetLabel(mLabelName)
    For i = 0 To mLlmClient1.FieldCount - 1
        Item = New ListViewItem("")
        ' get the field name from the control
        sFieldName = mLlmClient1.GetFieldName(i)
        ' get the max field length from the control
        iMaxFieldLen = mLlmClient1.GetFieldMaxLength(i)
        ' add the field name and max field length to the list view
        Item.SubItems.Add(sFieldName)
        Item.SubItems.Add(System.Convert.ToString(iMaxFieldLen))
        lstLabelData.Items.Add(Item)
    Next
    ' resize the list view columns to fit
    lstLabelData.Columns(0).Width = lstLabelData.Width / 3
    lstLabelData.Columns(1).Width = lstLabelData.Width / 3
    lstLabelData.Columns(2).Width = lstLabelData.Width / 3
End Sub
```

## JobName

### Declaration

```
Public Property JobName() As String
```

### Syntax

```
control.JobName = strJobName
```

### Description

The JobName property allows you to get or set the Job Name for the current job you are working on. This is only valid after calling GetLabel().

**Example**

```
Private Sub btnprint_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Btnprint.Click
    LoftClient.GetLabel("c:\Program files\Loftware Labeling\Labels\dotnet.lwl")
    LoftClient.PrinterNumber = 1
    LoftClient.JobName = "job2723"
    LoftClient.Quantity = 1
    LoftClient.Duplicates = 1
    LoftClient.Pages = 1
    LoftClient.SetData("text0000", TextBox1.Text) 'text0000 is the name of the field
    LoftClient.SetData(1, TextBox2.Text) ' 1 is the index number of the field
    LoftClient.PrintJob()
End Sub
```

## LabelDtd

**Note:** The LabelDtd call has been deprecated as of the Software Print Server version 10.0. This property will not function in version 10.0. See [XML Files](#) in this guide for more information.

### Declaration

```
Public Property LabelDtd() As String
```

### Syntax

```
control.LabelDtd = strDtdPath
```

### Description

The LabelDTD property allows you to get or set the current label.dtd path for the LPS. The label.dtd must exist in the same place on both the LPS and the Client in order for the XML creation to work properly.

**Note:** If you wish to modify any of the Software default xsd or dtd schema files that ship with the product, create and modify a renamed copy of the file. Otherwise, any subsequent install may overwrite changes you made.

#### Example

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Handles MyBase.Load
    LoftClient.Login("172.16.0.93", 2723)
    LoftClient.LabelDtd = ("c:\Program files\Software Labeling\batch\label.dtd")
End Sub
```

## LastErrorMessage

### Declaration

```
Public ReadOnly Property LastErrorMessage() As String
```

### Syntax

```
strLastError = control.LastErrorMessage
```

### Description

The LastErrorMessage property allows you to obtain the last error message for the control. These error messages are the same as the ones that are captured by the ErrorMessage Event.

#### Example

```
Private Sub btnlasterrmsg_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnlasterrmsg.Click
    TextBox3.Text = LoftClient.LastErrorMessage
    If (TextBox3.Text = "") Then
        TextBox3.Text = "There has been no error messages"
    End If
End Sub
```

## LoggedIn

### Declaration

```
Public ReadOnly Property LoggedIn() As Boolean
```

### Syntax

```
bLoggedIn = control.LoggedIn
```

### Description

The LoggedIn property allows you to check if you are logged into the LPS or not. Also takes socket connection into consideration.

#### Example

```
' logs the user in or out in response to clicking the login button
Private Sub btnLogin_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnLogin.Click
    ' if we're logged in, log us out
    If LlmClient1.LoggedIn() Then
        btnLogin.Text = "Login"
        LlmClient1.Logout()
        lstPrinters.Items.Clear()
        lstLabels.DataSource = Nothing
    Else ' otherwise, log us in
        btnLogin.Text = "Logout"
        LlmClient1.Login(IPAddress.Text, 2723)
        GetLabels()
        GetPrinters()
    End If
End Sub
```

## Pages

### Declaration

```
Public Property Pages() As Integer
```

### Syntax

```
control.Pages=1
```

### Description

The Pages property allows you to get or set the number of pages of labels printed. This refers to an entire copy of a page of labels when printing with layouts only.

**Example**

```
Private Sub btnprint_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Btnprint.Click
    LoftClient.GetLabel("c:\Program files\Software Labeling\Labels\dotnet.lwl")
    LoftClient.PrinterNumber = 1
    LoftClient.JobName = "job2723"
    LoftClient.Quantity = 1
    LoftClient.Duplicates = 1
    LoftClient.Pages = 1
    LoftClient.SetData("text0000", TextBox1.Text) 'text0000 is the name of the
        field
    LoftClient.SetData(1, TextBox2.Text) ' 1 is the index number of the field
    LoftClient.PrintJob()
End Sub
```

## PrinterName

### Declaration

```
Public Property PrinterName() As String
```

### Syntax

```
control.PrinterName = strPrinterName
```

### Description

Allows you to get or set the printer name (alias) of the printer to which you wish to print.

**Note:** The .NET control supports only one printer per print job. Appended jobs must be sent to the same printer as the first specified. All requests will be sent to the first printer specified.

**Example**

```
Private Sub btnprint_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Btnprint.Click
    LoftClient.GetLabel("c:\Program files\Software Labeling\Labels\dotnet.lwl")
    'LoftClient.PrinterNumber = 1
    LoftClient.PrinterName = "dotnetprinter"
    LoftClient.JobName = "job2723"
    LoftClient.Quantity = 1
    LoftClient.Duplicates = 1
    LoftClient.Pages = 1
    LoftClient.SetData("text0000", TextBox1.Text) 'text0000 is the name of the
        field
    LoftClient.SetData(1, TextBox2.Text) ' 1 is the index number of the field
    LoftClient.PrintJob()
End Sub
```

## PrinterNumber

### Declaration

```
Public Property PrinterNumber() As Integer
```

## Syntax

```
control.PrinterNumber=3
```

## Description

The PrinterNumber property allows you to get or set the LLM printer number of the printer to which you wish to print.

**Note:** The .NET control supports only one printer per print job. Appended jobs must be sent to the same printer as the first specified. All requests will be sent to the first printer specified.

### Example

```
Private Sub btnprint_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Btnprint.Click
    LoftClient.GetLabel("c:\Program files\Loftware Labeling\Labels\dotnet.lwl")
    LoftClient.PrinterNumber = 3
    LoftClient.JobName = "job2723"
    LoftClient.Quantity = 1
    LoftClient.Duplicates = 1
    LoftClient.Pages = 1
    LoftClient.SetData("text0000", TextBox1.Text)
    'text0000 is the name of the field
    LoftClient.SetData(1, TextBox2.Text) ' 1 is the index number of the field
    LoftClient.PrintJob()
End Sub
```

## Quantity

### Declaration

```
Public Property Quantity() As Integer
```

## Syntax

```
control.Quantity=1
```

## Description

The Quantity property allows you to get or set the total number of labels you want to print.

**Example**

```
Private Sub btnprint_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Btnprint.Click
    LoftClient.GetLabel("c:\Program files\Software Labeling\Labels\dotnet.lwl")
    LoftClient.PrinterNumber = 1
    LoftClient.JobName = "job2723"
    LoftClient.Quantity = 1
    LoftClient.Duplicates = 1
    LoftClient.Pages = 1
    LoftClient.SetData("text0000", TextBox1.Text)
        'text0000 is the name of the field
    LoftClient.SetData(1, TextBox2.Text) ' 1 is the index number of the field
    LoftClient.PrintJob()
End Sub
```

## Tray

### Declaration

```
Public Property Tray() As String
```

### Syntax

```
control.Tray = "Tray2"
```

### Description

The Tray property allows you to set the name of the tray from which the stock is taken for the labels you want to print (Windows printers and printers using the PCL driver).

**Example**

```
Private Sub Btnappend_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnappend.Click
    LoftClient.PrinterName = "dotnetprinter"
    LoftClient.JobName = "job2723"
    LoftClient.Tray = "tray2"
    LoftClient.Duplicates = 1
    LoftClient.Pages = 1
    LoftClient.SetData("text0000", TextBox1.Text) 'text0000 is the name of the
        field
    LoftClient.SetData(1, TextBox2.Text) ' 1 is the index number of the field
    LoftClient.Tray() 'Selects the Tray Number for the Printer for current job.
End Sub
```

## Software .NET Control Events

The Software .NET Control has three type of events:

- ErrorMessage Event
- WarningMessage Event
- InfoMessage Event

Each one signifies a different type of “error” message. Event handling is the primary and best way to handle errors when using this control for integration.

## ErrorMessage Event

### Declaration

```
Public Event ErrorMessage(ByVal strErrMsg As String, ByVal objSender As Object)
```

### Description

This event signifies some form of error has occurred. Both LPS errors (invalid logins, etc.) and printer error/critical failures may be returned through this event (which includes failure to generate proper XML for the LPS).

#### Example

```
Private Sub LoftClient_ErrorMessage(ByVal strErrMsg As String, ByVal objSender As
Object) Handles LoftClient.ErrorMessage
    MsgBox(strErrMsg)
End Sub
```

## WarningMessage Event

### Declaration

```
Public Event WarningMessage(ByVal strWarnMsg As String, ByVal objSender As Object)
```

### Description

This event is thrown in the case of a warning condition. These conditions are things that are brought to your attention but are not necessarily critical. Printed with Errors falls into this category.

#### Example

```
Private Sub LoftClient_WarningMessage(ByVal strWarnMsg As String, ByVal objSender As
Object) Handles LoftClient.WarningMessage
    MsgBox(strWarnMsg)
End Sub
```

## InfoMessage Event

### Description

This event is thrown for non-essential information. "Printed successfully" falls into this category.

#### Example

```
Private Sub LoftClient_InfoMessage(ByVal strInfoMsg As String, ByVal objSender As
Object) Handles LoftClient.InfoMessage
    MsgBox(strInfoMsg)
End Sub
```

## More Error Information and Return Codes

There are two other ways to handle errors in the Software .NET Control.

- **Return Codes** - Most of the methods used in the ActiveX Control return a code. Check these codes first.
- **LastErrorMessage property** - This property allows you to obtain the last set error message from either the Software Print Server or the Control.

### Return Codes in .NET

Return codes from functions are one way to check error conditions while using any kind of API. In the Software .NET Control return codes are typically Boolean (Yes/No). False always indicates failure. True always indicates success.

The return codes for other functions are specified as needed.

### LastErrorMessage

Whenever any type of critical error occurs within the Software .NET Control, the area that had the error calls OnErrorMessage, passing through an error string.

## Reference Table for the Software .NET Control

Name	Get	Set	Type	Comment
AppendJob	N/A	N/A	Method	Batches label requests together.
Clear Data	N/A	N/A	Method	Clears all the user-entered data from the current job
CurrentLabel	X		Property	Maps directly to the currently loaded label.
CurrentServer	X		Property	Name or IP address of connected LPS.
Duplicates	X	X	Property	Gets and Sets the # of duplicates for the current job.
ErrorMessage	N/A	N/A	Event	Signifies some form of error has occurred
FieldCount	X		Property	Gets the total # of fields in the currently loaded label.
GetFieldIndex	X		Method	Obtain the index of the field based on the field name.
GetFieldMax Length	X		Method	Takes index or name of fields, returns the max # of allowed chars.
GetFieldName	X		Method	Takes index of field and returns the name
GetLabel	X		Method	Downloads label info from LPS sets it as current loaded label



Name	Get	Set	Type	Comment
GetLabelList	X		Method	Returns list of available labels on the LPS.
GetLabelListExt	X		Method	Returns list of available labels and RFID tags with file extensions
GetLabelsPrinterID	X		Method	Passes in a label name (with path, optional), returns the ID of the printer the label was designed for.
GetPrinterByNumber	X		Method	Takes in the LLM Printer #, gives back name, alias, port, printer ID.
GetPrinters	X		Method	Obtains a list of all configured printers, returns as an integer.
KillJobs	N/A	N/A	Method	Removes all the jobs from memory, clears current job.
InfoMessage	N/A	N/A	Event	Thrown for non-essential information.
JobName	X	X	Property	Get or Set the current JobName for the current job.
LabelDtd	X	X	Property	Get or set the current label.dtd path for the LPS.
LastErrorMessage	X		Property	Obtain the last set error message from either the LPS or the Control.
Login	N/A	N/A	Method	Takes the Server Name or IP add. and port #, logs in as client to LPS.
LoggedIn	X		Property	Checks the user log in to the LPS
Logout	N/A	N/A	Method	Sends a logout request to the LPS and closes out connected socket.
Pages	X	X	Property	Get or set the number of pages of labels printed.
PrintJob	N/A	N/A	Method	Prints all of the current jobs
PrinterName	X	X	Property	Name of configured printer.
PrinterNumber	X	X	Property	Get or set LLM Printer Number to which you want to print.
Quantity	X	X	Property	Get or set the amount of labels to print.
SetData	X		Method	Takes name or index of field user wishes to use.
Tray	X	X	Property	Sets the tray number or ID for the printer stock (Windows printers and printers using the PCL5 driver)
WarningMessage	N/A	N/A	Event	Thrown for a warning condition.



---

## External Links

---

The following material was cited or consulted while developing LPS Family documentation or is a source of more detailed information that is beyond the scope of this documentation.

Apache. *[Apache Tomcat](http://tomcat.apache.org)*.

(<http://tomcat.apache.org>)

IETF. *[Internet Engineering Task Force](http://www.ietf.org)*.

(<http://www.ietf.org>)

Microsoft. *[How to disable IPv6 or its components in Windows](http://support.microsoft.com/kb/929852)*.

(<http://support.microsoft.com/kb/929852>)

Wikipedia. *[Comparison of web server software](http://en.wikipedia.org/wiki/Comparison_of_web_servers)*.

([http://en.wikipedia.org/wiki/Comparison\\_of\\_web\\_servers](http://en.wikipedia.org/wiki/Comparison_of_web_servers))