the professional labeling software

www.nicelabel.com, info@nicelabel.com

NiceWatch Enterprise User Guide

English Edition

Rev-1011

Euro Plus d.o.o.
Poslovna cona A 2
SI-4208 Šenčur, Slovenia
tel.: +386 4 280 50 00
fax: +386 4 233 11 48
www.europlus.si
info@europlus.si

Table of Contents

Introduction

Welcome to NiceWatch

NiceWatch is a middleware module that integrates label printing into an existing system (software applications, conveyor lines in the production areas, hardware devices like weight-scales and similar). NiceWatch is an event-based trigger application. It enables communication and synchronization of the label production process with certain events. If one or more of these events take place, the pre-defined action is triggered. You can control which actions and in what order are processed. NiceWatch product provides for the non-programming method of label printing integration.

NiceWatch is the entry level integration module for centralized label printing.

NiceWatch Enterprise is the enterprise solution for centralized integration of multi-threaded and high-volume label printing. NiceWatch Server is integral part of NiceWatch Enterprise product.

This instruction provides information for both products, NiceWatch and NiceWatch Enterprise. The functionality available only in NiceWatch Enterprise is clearly marked.

NiceWatch Enterprise Software Package

Note: This chapter applies to NiceWatch Enterprise only. **NiceWatch** is available with NiceLabel Suite edition.

Before you start installing the **NiceWatch Enterprise** software please check the content of the package to ensure it is complete. The package must include:

- NiceLabel 5 Enterprise Series CD
- User Guide
- Software Key number on a sticker (to activate NiceWatch Server)
- Hardware Key (to activate NiceLabel Pro for the label design)

Please contact your software vendor if your package misses any of the mentioned items.

Software Restrictions

All features explained in this User Guide are available in the NiceWatch Server application. However, some features are not available in the middleware product.

Note: NiceWatch Server is a software available of the NiceWatch Enterprise package from NiceLabelEnterprise Series of products. **NiceWatch** is a software of the NiceLabel Suite package from NiceLabelStandard Series of products.

Software Products

NiceLabel Software Products

NiceLabel is a family of professional labeling software products that brings a complete bar code printing solution and RFID Smart Label printing to desktop, mobile and enterprise users. Software offers an easy-to-use interface and meets any label design and printing requirement for efficient label printing solutions to users in retail, logistics, healthcare, chemical, automotive and other industries.

NiceLabel family of products offers the following:

- **Standard Series**
  - Represent the most complete range of "standard" labeling software
  - Easy to use but offers powerful functions expected from advanced label designers
  - Unique selling points: NiceForm, NiceWatch& Pocket NiceLabel
  - Tight control over printing process – NicePrintQueue & bi-directional drivers, reprint functionality
- **Enterprise Series** (Products for enterprises and toughest demands)
  - NiceLabel Print Center
  - NiceWatch Enterprise
- **Developer Series** (Integration solution for software publisher - "print engine")
  - NiceLabel SDK: SDK for software publishers, who need label printing capabilities in their software. NiceLabel SDK can be embedded in existing information systems or existing applications to provide support for label printing. NiceLabel SDK provides all label printing functionality of the software.
  - NiceLabel PocketSDK
  - NiceLabel WebSDK

NiceLabel Express

Wizard-based software meeting basic bar code labeling needs. The entry-level software includes many design elements of the NiceLabel Pro edition with the emphasis on simplified user interaction.

NiceLabel Pro

Full-featured software designed for professional label design and printing, including complete database support and ActiveX integration possibilities. A wide range of features

and options makes NiceLabel Pro a perfect and easy-to-use tool for any labeling requirement.

NiceLabel Pro includes the following modules:

- **NiceLabel Pro:** Main application for label design and printing.
- **NiceMemMaster:** Download fonts and graphics to the computer's memory card for optimized label printing.
- **NiceData:** Manage your databases.

NiceLabel Pro is available in Print Only edition as well.

NiceLabel Suite

NiceLabel Suite is a complete software solution for any kind of label design and print requirement. Multiple connectivity options allow users to perform stand-alone printing or integrate label printing into any network environment.

NiceLabel Suite offers interactive label printing capabilities such as integrating label printing into existing applications (ActiveX) or non-programming embedding of label printing to existing systems (NiceWatch).

NiceLabel Suite includes the following programs:

- **NiceLabel Pro:** Main application for label design and printing.
- **NiceForm:** Create your own custom designed data-entry applications without requiring any programming skills.
- **NiceWatch:** Integrate and automate label printing to any existing information system.
- **NiceMemMaster:** Download fonts and graphics to the printer's internal memory or memory card for optimized label printing.
- **NiceData:** Manage your databases.
- **NicePrint:** Manage fast and easy label printing.
- **NicePrintQueue:** manage your print jobs.

NiceLabel Suite is available in Print Only edition as well.

For detailed information about a particular program please refer to User Guide, available on the product CD and web site.

Pocket NiceLabel

Pocket NiceLabel is a program package for Windows CE that brings the power of label printing to portable Windows CE computers (Windows Mobile Device).

NiceLabel SDK

NiceLabel SDK is an ActiveX integrator edition of software developed for software publishers who need label printing capabilities in their software. NiceLabel SDK can be

embedded in existing information systems or existing applications to provide support for label printing. NiceLabel SDK provides all label printing functionality of the software.

NiceLabel Pocket PC Designer

NiceLabel Pocket PC Designer is a software package for desktop Windows computers that brings the power of label and form design to Windows mobile devices. After you have designed the required labels on the desktop PC, synchronize the labels with the Windows mobile device and print them from the Windows mobile device.

NiceLabel Print Center

NiceLabel Print Center is enterprise-level labeling solution with centralized control over all clients in the network. The enterprise edition manages the printing process centrally while label design and printing occurs locally on the client computers hosting NiceLabel Suite.

The NiceLabel Print Center software performs the following:

- Centralized print and event logging.
- Centralized control over printers and print queues (print jobs).
- Centralized logging of executed NiceWatch actions.
- Issuing alerts through email, NetSend, Instant Messenger, RSS feed, SMS or pager.
- Flexible licensing for clients.

The NiceLabel Print Center is comprised of two main components:

- NiceLabelEnterprise Print Manager, installed on server,
- NiceLabel Suite, installed on client computers.

The NiceLabel Print Center edition can be used with 32-bit and 64-bit Windows 2000 and Windows 2003 server families.

NiceWatch Enterprise

Centralized label printing middleware software that integrates the label printing process into enterprise-level products, such as Enterprise Resource Planning (ERP) systems, Warehouse Management Systems (WHS) and others.

The NiceWatch Enterprise performs the following:

- Multi-threaded label printing execution to ensure fast and stable operation in heavy-load environments (where a lot of print requests happen simultaneously).
- Remote administration to view list of triggers, start and stop triggers.
- Centralized logging of executed actions.

NiceWatch Enterprise includes the following modules:

- NiceWatch Server
- Enterprise Print Manager **(**EPM**)**

NiceWatch Enterprise also includes one NiceLabel Pro license. You will use NiceLabel Pro for label design.

The NiceWatch Server edition can be used with 32-bit and 64-bit Windows 2000 and Windows 2003 server families.

About this Manual

The NiceWatch Server User Guide helps you install and integrate this label printing solution into your enterprise IT environment.

The User Guide contains the following sections:

Introduction: Introduces you to NiceWatch and NiceWatch Enterprise software.

NiceWatch Server Overview: Learn about the NiceWatch Server architecture and understand the operation.

Installation and Activation: Learn how to install and activate the NiceWatch Server software.

Setting Up Application: Learn how to work with the user interface.

Working with Triggers: Learn how to set up and use Triggers, Data Filters, Variables and Actions.

Enterprise Print Manager: Discusses the module for remotely managing the NiceWatch Server.

Integration: Describes how the NiceWatch Server software can be integrated into your IT environment and existing applications.

Technical Support: Contacting technical support.

Typographical Conventions

Text that appears in **bold** refers to menu names, file names like **SETUP.EXE** and buttons like the **OK** button.

Text that appears in *italic* refers to confirming actions like *Read only*, locations like *Folder* or software window names.

Text enclosed in <Less-Than and Greater-Than signs> refers to keys from the desktop PC keyboard like <Enter>.

Variables are enclosed in [brackets] like [variable].

NiceWatch Enterprise

Overview

Today organizations use a wide variety of information systems, like enterprise resource planning or data warehousing solutions, where business-critical data is stored. Typically modifications of such systems are expensive or not possible at all. When labels need to be printed, organisations are seeking for labeling solutions capable to integrate into such environments with minimal or no changes to existing solution.

NiceLabel family of Enterprise products has a member specifically designed for such environments, called NiceWatch Enterprise.

NiceWatch Enterprise provides a true real-time, on-demand and automated label printing solution for any IT environment, while it's setup does not require any programming skills or changes in your IT systems.

NiceWatch Enterprise consists out of three components: NiceWatch Server, NiceLabelEnterprise Print Manager and NiceLabel Pro.

NiceWatch and NiceWatch Server

NiceLabel family offers the NiceWatch product in the Standard Series range. It differs from NiceWatch Server in a very important aspect: it is intended for single print environments, where only a limited number of requests happen simultaneously and there is no central log and alert possibilities with NiceLabelEnterprise Print Manager.

On the other hand, NiceWatch Server was built with a basic goal to create a robust solution for environments with several printers and numerous simultaneous print requests.

Functionality of the Middleware Integration Products

The middleware integration servers are an integration product capable of receiving label printing data by monitoring different resources such as a file, serial port, TCPIP port or e-mail message. A set of actions are defined which are triggered if predefined events happen. These actions can open a pre-designed label, connect it to a database, retrieve the variable values and print such filled label on a chosen printer. Here is a list of features:

- Automated bar code and RFID label printing.

- Bar code label printing to an unlimited number of printers on your enterprise-wide network.
- Bar code and RFID label printing to a network printer initiated by a Windows CE mobile device.
- Parse and map data from incoming structured or non-structured data streams (for example invoice file) to label variables.
- Import XML data from Oracle WMS and MSCA and print bar code and RFID label formats throughout the enterprise.
- Manipulate data from the triggers when an event occurs. Support for Visual Basic in included for demanding data-manipulation tasks.
- Support the incoming data from different types of triggers: text file, serial port, TCP/IP communication and data from emails.
- Provide support for multi-threaded high-volume printing with customizable number of global print engines, or local print engines per single trigger.*
- Log the printing activities to a central log server. Configure alerts in case of errors.*

Note: The abilities marked with asterisk character (*) are available only with the Enterprise Series.

NiceWatch Enterprise Business Connector support for SAP and HL7

Note: The support for SAP and HL7 is not available with regular middleware integration server. You need to purchase or upgrade to the advanced **NiceWatch Enterprise Business Connector** product. Please contact your nearest software reseller.

SAP Application support

For SAP-run businesses who need to print labels directly from SAP® NiceWatch Enterprise Business Connector is a certified device controller that directly connects to your data and prints it on any label printer.

Eliminating the need for programming, NiceWatch Enterprise Business Connector allows you to quickly choose your label design and data, and then print the label exactly as you wanted it. It focuses equally on supporting label printing through a direct native communication with SAP, enabling the user to print labels with SAP data from the SAP interface.

NiceWatch Enterprise Business Connector provides a bridge between the SAP system and the dynamic label printing operation, communicating with each side in its native language and maintaining complete transparency and ease of use.

HL7

HL7 (Health Level 7) is global messaging standard for healthcare applications and providers. The messaging interface enables healthcare, insurance and supplier institutions and software applications to communicate with each other in a common language.

NiceWatch Enterprise Business Connector communicates with HL7 v2.5 compliant software applications. The functionality is available for the management of outgoing HL7 messages and a HL7 filter is available for parsing incoming HL7 messages.

The HL7 standard has two main releases. The version 2.5 release is a linear, delimited data structured file that uses headers in the beginning of each segment to clearly separate the segments. The version 3 release of the standard is an XML data structured file. This release fixes the limitations of the v2 standard by allowing stronger type declaration of the fields and offering a more standardized structure of the result file sent in a message.

To begin, only the version 2.5 of the standard is supported in the software. According to our research, over 90% of the applications available today support version 2.5 of the HL7 standard only. Because of this, it is predicted that it will take 5 years or more to convert many of these version 2.5 deployments to version 3 XML. There are plans to enhance the software in the future to be HL7 version 3 compliant. Current version 3 customers can be serviced by using the XML filter in the software.

The overall system is one where the printer network is managed though the NiceWatch Enterprise Business Connector and the client stations are managed by the hospital Healthcare Information System (HIS), Laboratory Information System (LIS) or Pharmacy Information System (PIS).

Installation and Activation

Overview

Note: The installation and activation instructions in the next chapters describe how to install and activate the **NiceWatch Enterprise** product. To install and activate **NiceWatch** product please refer to the NiceLabel Suite user guide.

NiceWatch Enterprise installation consists out of two steps.

1. Install NiceWatch Server. You will use the middleware server to set up automatic label printing system that can optionally connect with NiceLabelEnterprise Print Manager.
2. Install NiceLabel Pro label designer. You will use NiceLabel Pro to design the labels.

You have received two installation CDs in the product package.

Use the *Enterprise Series CD* for NiceWatch Server installation. The same CD includes also installation for NiceLabel Pro edition. You need to install NiceLabel Pro as the label designer.

Use the second installation CD --*NiceLabel 5 Standard Series CD*-- for printer drivers (NiceDrivers), more user guides, technical papers and more information about NiceLabel software.

When you put any of the CDs in your CD-ROM drive, the demonstration program will start automatically. You can browse the directory of the CD, look at the brochures and technical documentation, sample files and other documents and install the software: NiceWatch Server and NiceLabel Pro for label design.

Note: You have to log on with Administrator rights to be able to perform the software installation.

System and Software Requirements

NiceWatch Enterprise

For successful NiceWatch Enterprise product implementation your equipment needs to fulfill predefined requirements. To install and run the NiceWatch Server module without the Enterprise Print Manager, you need at least the minimum configuration. For optimal performance and possibility to remotely manage the middleware server with EPM, please observe the recommended configuration.

Minimum configuration

- Computer with Intel or compatible Pentium class processor (1.0 GHz or faster recommended)
- At least 512 MB of RAM
- Hard disk with 1 GB of available disk space (actual consumed disk space depends on the amount of printing activity)
- CD-ROM drive
- Microsoft Windows XP with SP2

  Or

  Microsoft Windows 2000 Server with SP4

  Or

  Microsoft Windows 2003 Server with SP1
- Administrative rights to the local computer during installation.

Recommended configuration

For optimal performance and possibility to remotely manage your label-printing environment with EPM we recommend a solution with two computers. On the first you should install NiceWatch Server and the Enterprise Print Manager. EPM connects remotely and stores its data to the Microsoft SQL database on a second server. Following are recommended configurations for both computers.

Configuration for the **NiceWatch Server and EPM** server:

- Computer with Intel or compatible Pentium class processor (1.6 GHz or faster)
- 1 GB of RAM or more
- Hard disk with 1.2 GB of available disk space
- CD-ROM drive
- Microsoft Windows 2003 Server Edition, Web Edition, Enterprise Edition or Datacenter Edition; all with SP1
- Microsoft .NET Framework Version 2.0
- IIS6 installed
- A domain user account to access the database on a separate server is needed
- Administrative rights on the local computer during installation

Configuration for the **database** server:

- Computer with Intel or compatible Pentium class processor (1 GHz or faster)
- 1 GB of RAM or more
- Hard disk with 2 GB of available disk space (actual consumption depends on amount of printed labels and the EPM database archiving frequency)
- Microsoft Windows 2003 Server Edition, Web Edition, Enterprise Edition or Datacenter Edition; all with SP1

- Microsoft SQL Server 2005 Workgroup Edition, Standard Edition, Developer Edition

  Or

  Microsoft SQL Server 2000

Label Designer Single User Edition

To install and run the program you need at least the following:

- Computer with Intel or compatible Pentium III processor 500 MHz or higher processor (1 GHz or faster recommended)
- At least 128 MB of RAM
- Hard disk with 160 to 550 MB of free disk space (depending on the installation options)
- One of the 32-bit and 64-bit Windows operating systems starting with Microsoft Windows 2000, XP, 2003, Vista, 2008 or 7
- CD-ROM drive
- Administrator rights to the local computer during installation

---

Installing NiceWatch Enterprise

Run Middleware Integration Server Installation

Use the installation wizard to install NiceWatch Enterprise product on your server. To install the NiceWatch Enterprise software, do the following:

1. To ensure complete installation, close all open software applications before you start installing the NiceWatch Enterprise software.
2. Put the NiceLabel 5 Enterprise Series CD in your CD-ROM drive. An installation window will open automatically.

Note: If the installation wizard does not start automatically, go to the main CD directory of your NiceLabel 5 Enterprise Series CD and double-click on the file START.EXE.

3. Select **Install Enterprise Products** and click on **NiceWatch Enterprise** in the next screen.
4. The installation of NiceWatch Enterprise will start.
5. Follow the on-screen prompts. There are two important steps during the installation.
6. Make sure you allow the NiceWatch Server and Label Services to communicate through your firewall. The status of label printing and similar useful information are sent to the Enteprise Print Manager (EPM). If you do not open the

communication for these two applications, your EPM will not display all data and you will not see the status of your printers.

7. You must tell NiceWatch Server if and which EPM it should use. EPM is a web-based application that provides remote overview into the NiceWatch Server functionality. You can use it to review the middleware server configuration, set alerts for case of problems and see the printing logs.

   The following options are available:

   - **Do not use EPM:**NiceWatch Server will not connect to the EPM. It will run in a standalone mode, it will have a local log file and there is no alerting possible. In this case you activate NiceWatch Enterprise product using the Software Key Number.
   - **Install EPM on this computer:** Next to NiceWatch Server you will install also the EPM on the same computer. Make sure your system meets the requirements for EPM install. In this case you activate NiceWatch Enterprise product through the EPM.

     Note: Refer to the NiceLabelEPM installation guide how to install EPM.

   - **I will install EPM on another server:** You will install the middleware module on the current computer, but EPM on some other computer. The folder that contains EPM installation will be shared on your computer as <COMPUTER_NAME>\EPMSETUP. You must start the EPM installer from the other computer using the installer in the shared folder from your computer. In this case you activate NiceWatch Enterprise product via the EPM.

     Note: Refer to the NiceLabelEPM installation guide how to install EPM.

   - **Use already installed EPM:** If you already have NiceLabelEPM set up on some server, you can simply connect your NiceWatch Server to the existing EPM. One EPM can handle multiple NiceWatch Server installations. Just enter the computer name that has the EPM already installed.

   Click on **Next**, when you have decided if and where to install NiceLabelEPM.

8. Click on **Install** to start with NiceWatch Server installation.

Installing NiceLabel Pro

Run Label Designer Installation

Use the installation wizard to install the label designer on your server. To install the NiceLabel Pro software, do the following:

1. To ensure complete installation, close all open software applications before you start installing the NiceLabel Pro software.
2. Put the NiceLabel 5 Enterprise Series CD in your CD-ROM drive. An installation window will open automatically.

Note: If the installation wizard does not start automatically, go to the main CD directory of your NiceLabel 5 Enterprise Series CD and double-click on the file START.EXE.

3. Select **Install Enterprise Products** and click on **NiceLabel Pro/Suite** in the next screen.
4. The installation of NiceLabel Pro will start.
5. Follow the on-screen prompts. Make sure to select NiceLabel Pro when prompted to select the installation type between NiceLabel Pro and NiceLabel Suite.

For more details about NiceLabel Pro installation refer to the NiceLabel Pro User Guide available on NiceLabel5 Enterprise CD (Documentation -> User Guides -> NiceLabel Pro).

Label Services

A service called Label Services (LblServices.EXE) is installed and started on each client after NiceLabel software installation. Label Services also runs a sub-process NDH.EXE. Both services are monitoring the local printer and its print job status, which results in the local job log.

Printer and job statuses are also sent to the Enterprise Print Manager (EPM), where the central job log is updated. It contains accurate information about the printed jobs and printer statuses.

Note: Label Services needs to run so labeling client can report print jobs to the EPM.

If you stop the Label Service, the following functionality will not be available:

- Advanced Printing - feedback from windows print spooler about print job progress is not written to the log file.
- Communication between the client and EPM is not active (sending print job/event information from the client to the Print Center).
- NicePrintQueue cannot be used.

If any of the mentioned functionality is not available, the first thing to check is the status of the Label Services on the workstation.

To check the status of Label Services, do the following:

1. Open **Control Panel -> Administrative Tools**.
2. Double-click the **Services** icon.
   The list of all registered services of your Windows system will open.
3. To start the **Label Services** service, right-click on the Label Services service and select **Start** from the context menu. The service status will change to **Started**.

Activating NiceWatch Enterprise

Overview

You can start using the NiceWatch Enterprise software immediately after installation. However, it will start and run in Demo mode, until you activate it.

You can activate NiceWatch Enterprise automatically through the Activation website (recommended) or manually through the activation Key Number Activation Web site. You would only use the manual method, if the computer with NiceWatch Enterprise does not have access to the Internet.

Before you start the activation process, get familiar with the following terminology:

- Key number: The sequence of 25 characters provided with the NiceLabel software.
- Registration number: Unique number generated during the activation process. The registration number validates the entered software key.
- Activation code: Issued on the Activation web site. The activation code activates the NiceLabel software on your computer.

Activate the Middleware Integration Module

Before you activate NiceWatch Server make sure, you have your 25-digit Key Number at hand. To activate the software you will have to enter your contact data and validate the Key Number on the internet activation website. Activation is one-time process.

Note: There are two types of NiceWatch Server activations. The first, if you use Enterprise Print Manager (EPM), and the second, if you do not use the Enterprise Print Manager, but install NiceWatch Server as a stand-alone application.

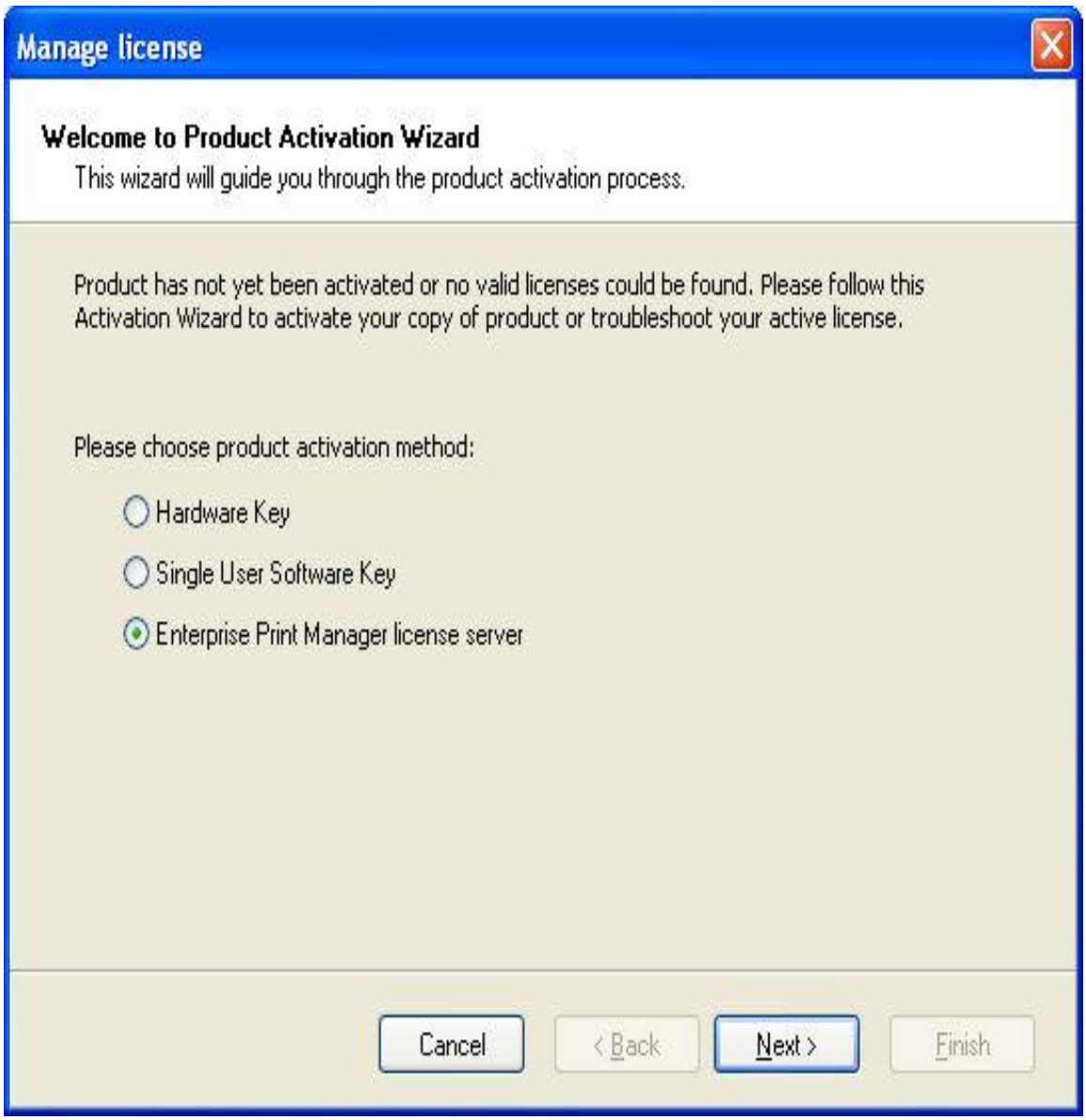
Activate Middleware Integration Module using the EPM

To activate NiceWatch Enterprise using the EPM, do the following:

1. Run NiceWatch Server. The *Manage License* window will open.

Note: If you do not complete the activation on the first run, on each next run you will have to click on the button **Manage License** to open the *Manage License* window.

2. Select the option **Activate the package** in the *Manage License* dialog box. Click on **Next**.
3. Select the option **Enterprise Print Manager** license server. Click on **Next**.

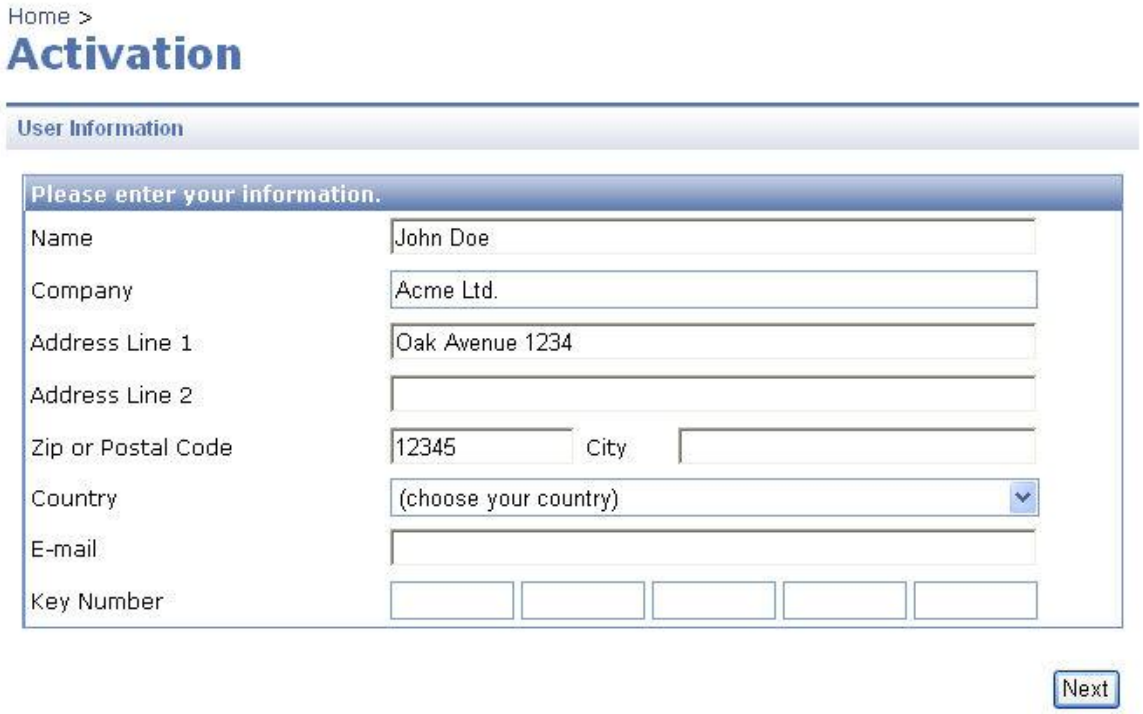


Selecting Enterprise Print Manager license server activation method

4. Type in the server name of the computer with installed EPM. Click on **Next**.
5. If you have entered the correct server name, you will see the *Enterprise Print Manager license server* dialog box informing you that events and print logs will be saved to the EPM.
6. Click on **Finish** to close the *Manage License* dialog box.

If the Enterprise Print Manager hasn't been activated yet, you will see a warning message about EPM state. Select the option to activate the EPM, then do the following:

1. Open NiceLabelEPM.
   Point your internet browser to the http://*server*/EPM/Home.aspx where the *server* represents the computer name, where you installed EPM.
2. Click on the **Licensing** tab.
3. Click on the **Activate new product** button. The *Activation - User Information* page will open.
4. Enter the required customer information and the key number that was delivered with the NiceWatch Enterprise product.



Entering user information and the Key Number

5. Click on the **Next** button. The *Product Activation* page opens and the Registration number is generated automatically. If your computer has an Internet connection, you can complete the activation by clicking the **Finish** button.
6. The activation website will be contacted and the software will be activated.

If the computer where you started the activation procedure does not have Internet access, do the following:

1. Write down the Key number and the Registration number.
2. Write down the Internet activation web address.
3. Go to a computer with Internet access and enter the Activation web page address in the Internet Explorer.
4. Enter the Key number and the Registration number. Enter other user information when prompted.

5. Click on the **Activate** button to retrieve the Activation code. Write it down.

Note: The Key and Registration numbers, the Activation code and information in other fields are case-sensitive, so be careful when writing them down.

6. Go back to the computer where the Product Activation page is still open.
7. Enter the Activation code.
8. Click on the **Finish** button and follow the prompts to complete the activation process.

NiceWatch Manager must not use a second license, when server is running under a different user account

Once the EPM is activated, you can connect your NiceWatch Server to it, acquire the necessary license and activate NiceWatch Server as well. Just start from step 1 at the top of this topic.

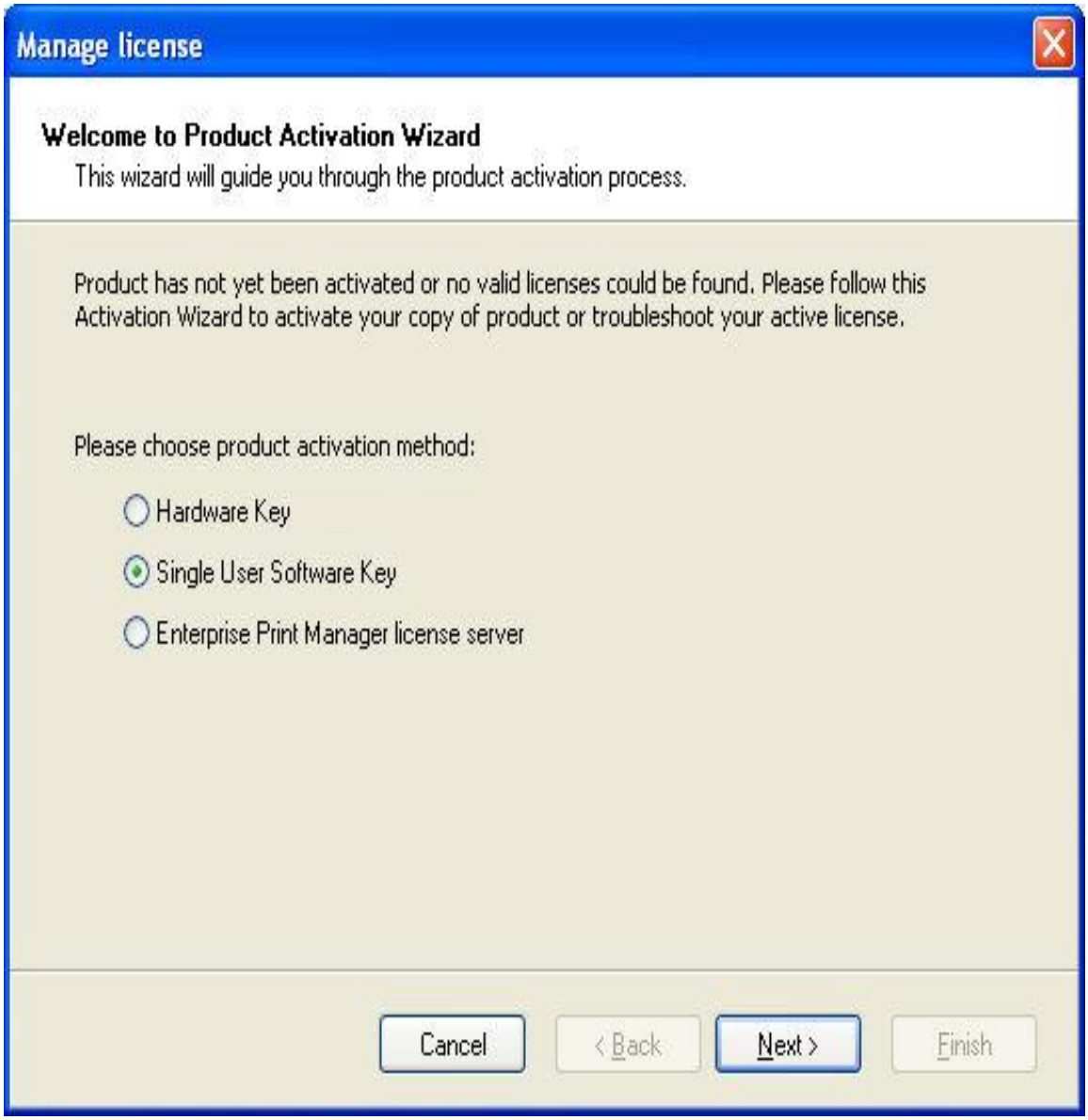
Activate the Middleware Integration Module as a stand-alone product

To activate NiceWatch Enterprise as a stand-alone product, do the following:

1. After the installation is complete, run NiceWatch Server. The *Manage License* window will open.

Note: If you do not complete the activation on the first run, on each next run you will have to click on the button **Manage License** to open the *Manage License* window.

2. Select **Activate the package** in the *Manage License* dialog box and click on **Next**.



Select Single-User Software Key activation method

3. Select **Single-user Software Key** and click on **Next**.
4. Enter the required customer information and the key number.
5. Click on the **Next** button. The window *Manage License – Activation Code* will open.
6. To activate the software automatically, click on the link **Connect to the Activation server** (recommended).
7. To activate the software manually, click on the Internet link for Web page activation. Your Internet browser will open the Key Number Activation Web site. You must enter at least the field marked with the asterisk (*) character.

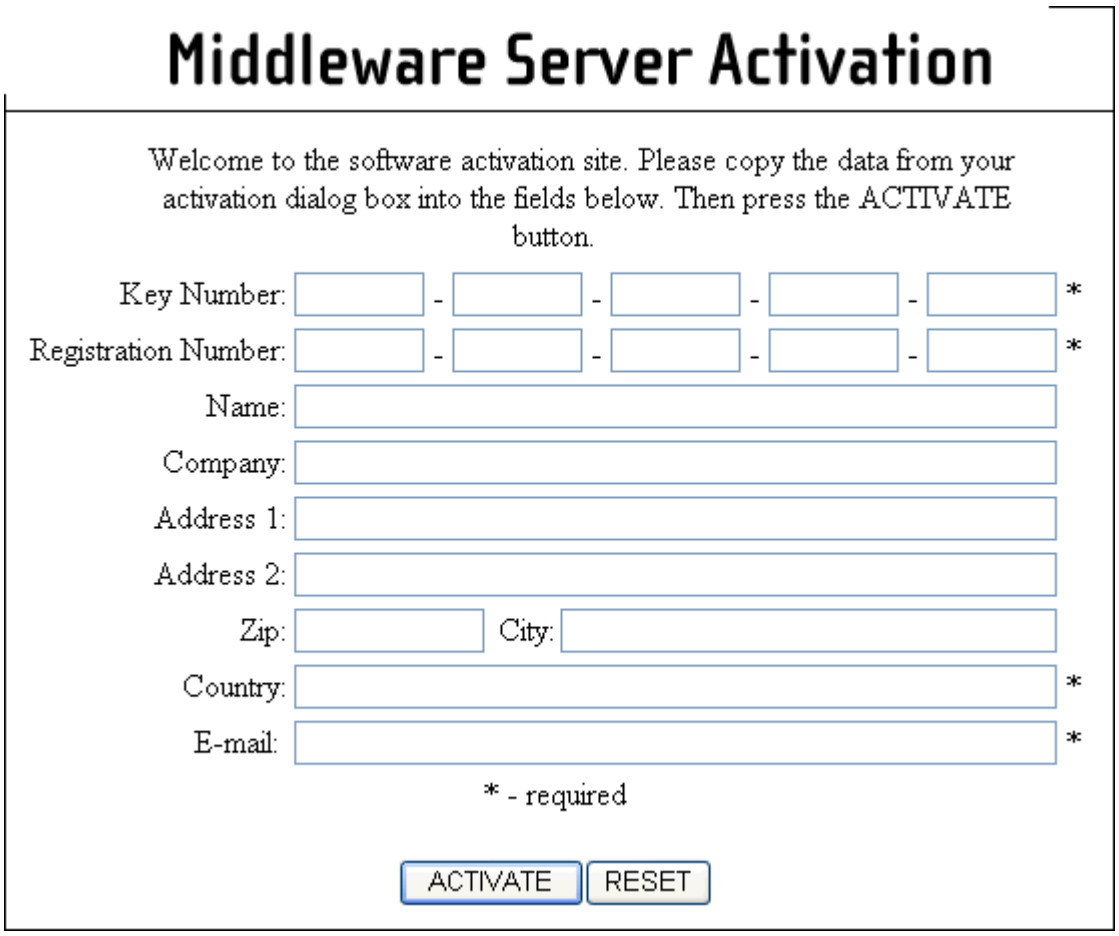
- Enter the key number and the registration number. Enter other user information, if prompted.
- Click on the **Activate** button to retrieve the Activation code.
- Copy the Activation code and paste it in the field **Activation Code** in the open *Manage License – Activation code* window.

7. Click on the **Next** button and follow the prompts to complete the activation process.

Note:
You can still activate the software if the computer does not have access to the Internet.

If the computer where you installed software does not have Internet access, do the following:

1. Follow the steps one (1) to four (4) from the previous section.
2. Write down the Key number and the Registration number.
3. Write down the Internet activation Web address.
4. Go to a computer with Internet access and open the activation Web site in the Internet Explorer.
5. Enter the Key number and the Registration number. Enter other user information when prompted.



Internet activation website

6. Click on the **Activate** button to retrieve the Activation code. Write it down.

Note:
The Key and Registration numbers, the Activation code and information in other fields are case-sensitive, so be careful when writing it down.

7. Go back to the client computer where the window Manage License is still open.
8. Enter the Activation code.
9. Click on the **Next** button and follow the prompts to complete the activation process.

Activate the Label Designer

You have to activate the client software if you want to use it. If you do not complete the activation process, the label designer will run in demo mode and will be limited in functionality.

The *Product Activation Wizard* will always run after the client installation. If you want to change the activation type, you can also start activation process from the label designer software. Select **Help -> Manage License** in the label designer.

The label designer client that ships with the NiceWatch Enterprise is activated with the enclosed Hardware Key. Please refer to the activation chapter in the Quick Start Guide or the User Guide.

Upgrading Licenses

Upgrade the Number of Licenses

Note: This chapter applies to NiceWatch Enterprise edition only.

By default NiceWatch Enterprise product ships with the license for one NiceWatch Server instance. If you have higher label printing needs, you can increase the number of NiceWatch Server licenses.

You can only upgrade the number of licenses, when you activate your NiceWatch Server from the Enterprise Print Manager. If you activate the NiceWatch Server as a stand-alone application, no upgrade is possible.

To upgrade the number of licenses, do the following:

1. Open Enterprise Print Manager.
   Point your internet browser to the http://*server*/EPM/Home.aspx where the *server* represents the computer name, where you installed EPM.
2. Click on the **Licensing** tab.
3. Click on the **Activate new product** button. The *Activation - User Information* page will open.
4. Enter the required customer information and the key number that was delivered with the product.
5. Click on the **Next** button. The *Product Activation* page opens and the Registration number is generated automatically. If your computer has an Internet connection, you can complete the activation by clicking the **Finish** button.
6. The activation website will be contacted and the licenses will be activated.
7. Now you can install additional servers and activate them with licenses on the EPM. Refer to the chapter **Activate NiceWatch Server using the EPM**.

Upgrade the Functionality

Note: This chapter applies to NiceWatch Enterprise edition only.

The classic NiceWatch Enterprise edition provides benefits over the standard NiceWatch application in views of high-load enterprise printing (multiple print engines) and central log server (Enterprise Print Manager). But NiceWatch Enterprise can offer even more.

If you require some extra functionality not covered by NiceWatch Enterprise out-of-the-box, please contact your software reseller. The functionality you are looking for might already be available with some NiceWatch Enterprise add-on.

To upgrade your NiceWatch Enterprise license to some new feature, do the following:

1. Start NiceWatch Server Manager.
2. Select Help -> Manage License.
3. Enable the option **Manage current software key license**. Click on **Next**.
4. Select the option **Upgrade software key license**. Click on **Next**.
5. Enter the **Upgrade Key Number** and follow on-screen instructions.

Generally speaking, you must validate your Upgrade Key Number on the internet activation website. The recommended method is to simply click the option **Connect to the Activation server**.

Using License Manager

Note: The information in this topic applies to Enterprise Series only.

The program License Manager is used to monitor and manage licenses. License Manager displays the information of the current client and also provides an overview of the total license occupation. The Enterprise Print Manager (EPM) stores the licenses for the NiceWatch Server.

You can also use it to enable/disable off-line mode for workstations. Off-line mode enables selected workstations to function in spite of the fact that Enterprise Print Manager (EPM) is not available or not accessible. It is used in environments where mobile label printing is required or in events where server system failure would otherwise stop the label production.

Force Off-line Mode: NiceWatch Server will not check for the presence of the EPM. The application will automatically start in the off-line mode. When the license lease expires it will stop running and you must renew the license by switching off this option and re-connecting to the EPM.

Note:
The off-line license expires after 15 days.

Setting Up the Application

User Interface

Basic and Advanced User Interface

The middleware integration software has two-level user interface. By default, basic user interface is enabled that makes the software easier to use for beginners and inexperienced users.

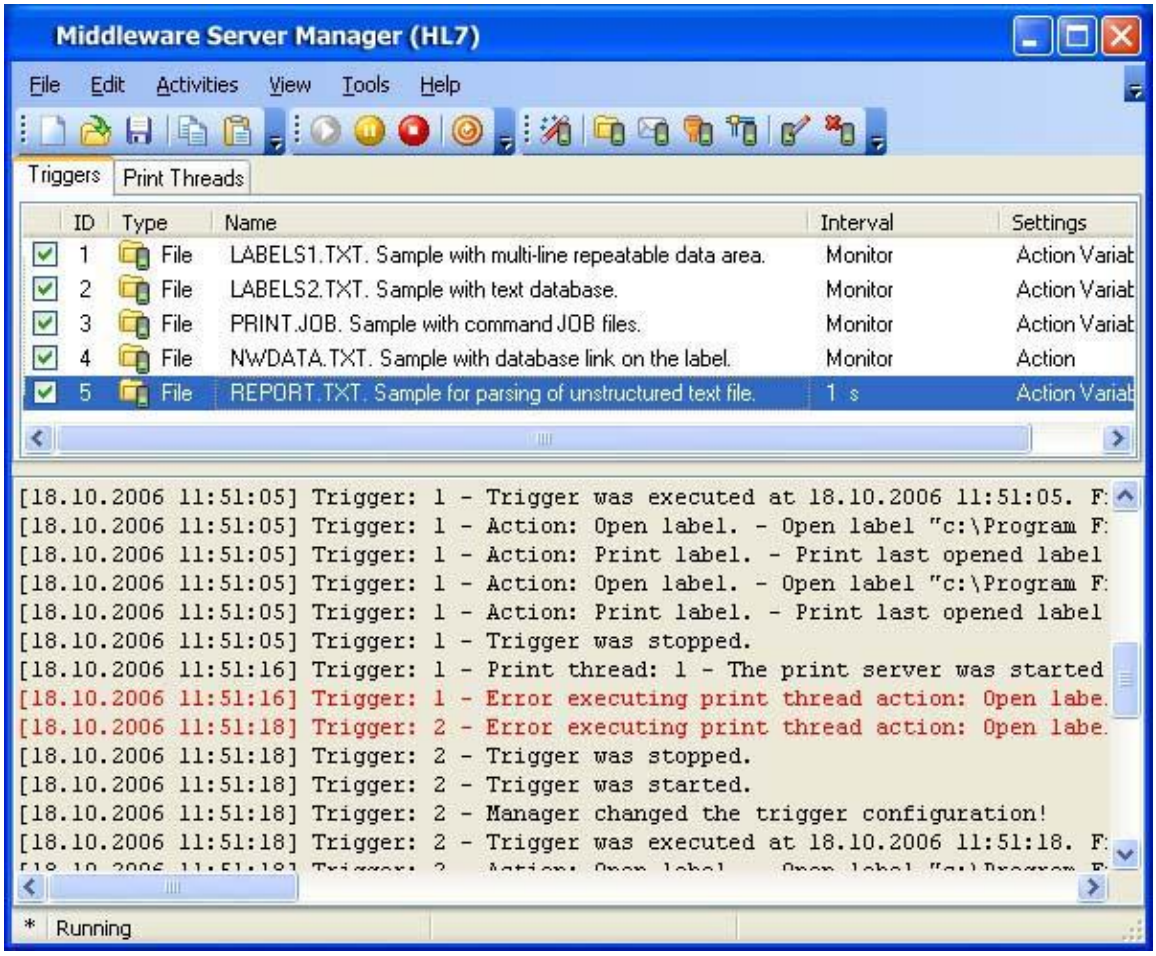
To switch to the Advanced mode showing all program functionality, click on the **Advanced** button in the bottom of some dialog boxes.

For example:
The dialog box **Actions Editor** has two-level user interface.

Main Window

When you start the middleware integration module the main window opens that consists of several parts. Click on the part of the client window to see the description.

Note: Some parts of the main window are accessible with the NiceWatch Server only.



Print Thread

Click on the **Print Thread** tab to see the usage and load of the print engine. Normally, the print engine handles all jobs as they appear, and there should be none queued in the thread. If a queue builds up and persists, the system is working above capacity and you may need to upgrade it.

Customizing the Software

To change the preferences of the application, do the following:

1. Select **Options** command in the Tools menu.
   The **Options** dialog box opens.
2. You can change the options regarding the printing and logging of activity.

To customize the menus and toolbars, do the following:

1. Select **Customize** command in the Tools menu.
   The **Customize** dialog box will open.
2. Make modifications to the settings.
3. Click on the **OK** button.

Note:
You can also change the positions of the toolbars on-screen. Just drag them to the new position.

Working with Triggers

Defining Triggers

Overview

The middleware integration server is an event-based application and monitors different system events. When the application detects a change in an event, predefined actions are triggered. The data that triggers the action can be used to start the label printing process alone or can contain the values for the label variables.

The middleware integration server offers four trigger options.

Define Trigger with Multiple Print Engines

Note: This chapter applies to middleware integration server from the Enterprise Series. Only NiceWatch Server (integral part of NiceWatch Enterprise) has support for multi print engines. The regular middleware integration server can use one print engine.

By default NiceWatch Server uses only one print engine for all triggers. It depends on the label printing demand. For some users one print engine is enough, for high-load scenarios using multiple print engines can be very handy. The system becomes more responsive and copes with the printing demands easier.

You have two options when choosing multiple print engines.

- You can define more global print engines that are used from all triggers.
- You can define private print engines that apply only to the selected trigger. These print engines cannot be used outside of that one trigger. The print engines are reserved to the selected trigger that needs high label throughput.

To define more global print engines, do the following:

1. Start the middleware integration Manager.
2. Select **Tools -> Options**.
3. Go to NiceWatch Server section, select **Print Thread**.
4. Increase the option **Number of global print threads**. Each new print engine is run in its individual thread and will be available to all triggers that do not have defined any private print engines.

Note: Be careful when increasing the number of print engines. Each print engine takes its own amount of RAM. Make sure you have enough free RAM. Set the number of print engines to some lower number and verify the consumption of your memory. You can use Windows Task Manager.

To define more private print engines per trigger, do the following:

1. Start middleware integation server Manager.
2. Define your trigger.
3. When ready, go to the **Preferences** tab.
4. Enable the option **Use local print threads**.
5. Increase the option **Number of local print threads**. Each new print engine is run in its individual thread and will be available to this trigger alone.

Note: Be careful when increasing the number of print engines. Each print engine takes its own amount of RAM. Make sure you have enough free RAM. Set the number of print engines to some lower number and verify the consumption of your memory. You can use Windows Task Manager.

Using File Trigger

What is File Trigger?

A file trigger event occurs when a monitored file or a file in a monitored folder changes (the file modification time-stamp changes). For example, the computer running the middleware integration server is connected to the same network as the enterprise ERP server that runs the main corporate database. When a label needs to be printed, the ERP application on the server exports the part of a database that holds the required label data. Those data fields are stored in an ASCII text file on a disk that is shared on the network.

Middleware integration server detects the new file or the change of the file and initiates label printing. Variable values are extracted from the text file and transferred to the label where they are used at print-time.

The monitored trigger file can be the following:

- "Dummy" file containing no data that only starts a defined action.
- Text file containing structured text database or XML data.
- Text file containing unstructured data.
- .JOB file containing NiceCommands.

Using File Trigger

To define the trigger do the following:

1. Select the command **Add Trigger** from the Edit menu.
2. Select the command **File**.
   The properties of the new File trigger will open.

Note:
You can also click on the icon  in the trigger toolbar.

4. Go to the **File** tab.
5. Define the properties of the files you will monitor..

6. Go to the **Action** tab.
7. Define the actions that should be executed, when the trigger occurs.
8. Click on the **OK** button.
   The trigger is ready for usage.

If the File trigger provides data values for the labels, you will have to extract the values from the incoming data and transfer them to the variables. Do the following:

1. Double click the File trigger to open its properties.
2. Go to the **Variables** tab.
3. Define the variable that exist on the label.

Note:
You can manually define the variables or import the from the label file. If creating variables manually, make sure to use the same names as are defined in the label.

4. Go to the **Filter** tab.
5. Tick the option **Use filter** to enable filter functionality.
6. Select the appropriate **Filter type** that matches the format of the incoming data.
7. Define the fields in the incoming data stream.
8. Link the fields with the variables.
9. Click on the button **Verify the Definition** to test the processing of your filter.
10. Click on the **OK** button.

Using Email Trigger

What is Email Trigger?

The middleware integration server checks an email on any POP3 mail server. When an email is received, an email event is triggered and pre-defined actions are executed. The subject of the email message determines if the event should be triggered or not. The body of the message can hold both the data for variable values and NiceCommands that specify label printing procedures in detail.

The email trigger is an effective way for two or more locations not connected by a Virtual Private Network (VPN) to safely and easily send the data to each other in order to print labels remotely. Firewall concerns are eliminated with a plain text-based email message.

Note: The personal email account information included in the trigger file is securely encrypted.

Define Email Trigger

To define the trigger do the following:

1. Select the command **Add Trigger** from the Edit menu.
2. Select the command **Email**.
   The properties of the new Email trigger will open.

Note:
You can also click on the icon  in the trigger toolbar.

4. Go to the **Email** tab.
5. Define the properties of the email trigger.
6. Go to the **Action** tab.
7. Define the actions that should be executed, when the trigger occurs.
8. Click on the **OK** button.
   The trigger is ready for usage.

If the Email trigger provides data values for the labels, you will have to extract the values from the incoming data and transfer them to the variables. Do the following:

1. Double click the Email trigger to open its properties.
2. Go to the **Variables** tab.
3. Define the variable that exist on the label.

Note:
You can manually define the variables or import the from the label file. If creating variables manually, make sure to use the same names as are defined in the label.

4. Go to the **Filter** tab.
5. Tick the option **Use filter** to enable filter functionality.
6. Select the appropriate **Filter type** that matches the format of the incoming data.
7. Define the fields in the incoming data stream.
8. Link the fields with the variables.
9. Click on the button **Verify the Definition** to test the processing of your filter.
10. Click on the **OK** button.

Using COM Port Trigger

What is Serial (COM) Port Trigger?

The middleware integration server captures data received from a serial (COM) port and triggers an action. The application can receive different formats of incoming data. For example, a bar code scanner attached to the serial port scans the bar code for data. The application accepts the input data and initiates label printing. The print engine opens the label and pulls a specific record from a database that complies with the scanned input data. A similar scenario is also valid for other types of serial port devices such as weight scales.

The application can also communicate with a serial device through Visual Basic script functions. The received data can be parsed using a filter or further manipulated by using Visual Basic action.

Define COM Port Trigger

To define the trigger do the following:

1. Select the command **Add Trigger** from the Edit menu.
2. Select the command **COM Port**.
   The properties of the new COM Port trigger will open.

Note:
You can also click on the icon  in the trigger toolbar.

4. Go to the **COM Port** tab.
5. Define the properties of the serial port.
6. Go to the **Action** tab.
7. Define the actions that should be executed, when the trigger occurs.
8. Click on the **OK** button.
   The trigger is ready for usage.

If the COM Port trigger provides data values for the labels, you will have to extract the values from the incoming data and transfer them to the variables. Do the following:

1. Double click the COM Port trigger to open its properties.
2. Go to the **Variables** tab.
3. Define the variable that exist on the label.

Note:
You can manually define the variables or import the from the label file. If creating variables manually, make sure to use the same names as are defined in the label.

4. Go to the **Filter** tab.
5. Tick the option **Use filter** to enable filter functionality.
6. Select the appropriate **Filter type** that matches the format of the incoming data.
7. Define the fields in the incoming data stream.
8. Link the fields with the variables.
9. Click on the button **Verify the Definition** to test the processing of your filter.
10. Click on the **OK** button.

Using TCP/IP Trigger

What is TCP/IP Trigger?

The command for starting label printing or the label data itself can be received over any specified TCP/IP port number. Mobile devices that are performing real-time data collection can send label requests over the same wireless infrastructure and print labels to any printer on your enterprise network.

For example, a portable handheld computer is used in the company's warehouse for data collection. The handheld has a built-in bar code scanner. The bar code is scanned to

Pocket edition of the labeling software, which processes the label data, prepares a batch command file (.JOB file) and transmits the data over TCP/IP protocol to middleware integration application running on the server. Based on the data scanned from the bar code the print engine queries the enterprise database and prints the label with the specified data.

The TCP/IP socket trigger enables mobile wireless data collection devices or UNIX applications to send label requests to middleware integration server. UNIX applications do not need to use FTP or Telnet to drop a file to a Windows server in order to trigger a label printing event.

Define TCP/IP Trigger

To define the trigger do the following:

1. Select the command **Add Trigger** from the Edit menu.
2. Select the command **TCP/IP**.
   The properties of the new TCP/IP trigger will open.

Note:
You can also click on the  icon in the trigger toolbar.

4. Go to the **TCP/IP** tab.
5. Define the properties of the TCP/IP server.
6. Go to the **Action** tab.
7. Define the actions that should be executed, when the trigger occurs.
8. Click on the **OK** button.
   The trigger is ready for use.

If the TCP/IP trigger provides data values for the labels, you will have to extract the values from the incoming data and transfer them to the variables. Do the following:

1. Double click the TCP/IP trigger to open its properties.
2. Go to the **Variables** tab.
3. Define the variable that exist on the label.

Note:
You can manually define the variables or import them from the label file. If creating variables manually, make sure to use the same names as the names defined in the label.

4. Go to the **Filter** tab.
5. Tick the option **Use filter** to enable filter functionality.
6. Select the appropriate **Filter type** that matches the format of the incoming data.
7. Define the fields in the incoming data stream.
8. Link the fields with the variables.
9. Click on the button **Verify the Definition** to test the processing of your filter.
10. Click on the **OK** button.

Using Database Trigger

What is a Database Trigger

The Database Trigger periodically checks a defined database, if there are new or updated records available in user-defined intervals. Four types of database triggers are supported:

- **Get records based on a unique incremental value**: This option will require the table to have key (autoinc) field. NiceWatch will internally remember which value was last processed and will only select records that were not processed yet. After new records are processed by NiceWatch, NiceWatch will remember the value of the key field of the last processed record. The user will have to set the table and name of the key field.
- **Get records and delete them**: The user will only have to set table. All records returned by this SQL will be processed by NiceWatch. After each record is processed, the record will be deleted from the table.
- **Get records and update them**: The user will have to set the table, update field and update value. All records returned by this SQL (all records where the update field value is different than the update value) will be processed by NiceWatch. After each record is processed, the update field of that record will be set to the update value, so that the next time this same record is not returned.
- **Get and manage records with custom SQL**: The user will be able to write a custom SQL to get records and update records.

Using the Database Trigger

To define the trigger, do the following:

1. Select the command **Add Trigger** from the Edit menu.
2. Select the command **Database**.
3. The properties of the new database trigger will open.

Note:
You can also click on the Database Trigger icon in the trigger toolbar.

4. On the **Database** tab, define the **Database Connection** and **Execution Event**. For a basic description of events, see the **What is a Database Trigger** topic.
5. Set the database connection and update settings in **Execution Options**.

Go to the **Action** tab.
Define the actions that should be executed, when the trigger occurs. Click on the **OK** button.
The trigger is ready for use.

If the database trigger provides data values for the labels, you will have to extract the values from the incoming data and transfer them to the variables. Do the following:

1. Double click the database trigger to open its properties.

2. Go to the **Variables** tab.
3. Define the variable that exist on the label.

Note:
You can manually define the variables or import them from the label file. If you are creating variables manually, make sure to use the same names as the names defined in the label.

4. Go to the **Filter** tab.
5. Tick the option **Use filter** to enable filter functionality.
6. Select the appropriate **Filter type** that matches the format of the incoming data.
7. Define the fields in the incoming data stream.
8. Link the fields with the variables.
9. Click on the button **Verify the Definition** to test the processing of your filter.
10. Click on the **OK** button.

Note:
If you do not set a filter and connect it to label variables, the trigger will not be able to print out more than one label in cases where several records are retrieved.

Custom SQL Statements

The use of advanced Custom SQL Statements is available with the **Get and manage records with custom SQL** event type. Click the **Define...** button to access these advanced settings.

The user is able to write custom SQL commands, used to get and update records. SQL uses parameters for this operation.

Examples:

Get values from the Articles table for all records, where the "printed" value is set to false (the record was not printed yet).

```
SELECT articleID, articlename, printed FROM articles
WHERE printed = false
```

The trigger processes each of these records and calls the bottom query.

```
UPDATE articles
SET printed = true
WHERE articleID = :articleID
```

It is possible to use parameters, which are fields from the first query. In this case, the parameter *:articleID* would return the Article ID of the record, which was processed.

For databases that do not support deleting or updating records, only the Key field type database trigger is possible (Excel for example). All field names from the source data

table are allowed to be used in UpdateSQL as parameters, which means that the values from the source record will be used in Update SQL.

Using Web Service Trigger

What is the Web Service Trigger

Web service triggers are triggers, which create a SOAP server on which server clients can execute methods. When a method is called by a Web Service, the trigger is executed with the preset parameters.

Note: This functionality is only available with the NiceWatch Enterprise Business Connector product.

Define a Web ServiceTrigger

To define the trigger do the following:

1. Select the command **Add Trigger** from the Edit menu.
2. Select the command **Web service**.
   The properties of the new Web service trigger will open.

Note:
You can also click on the  icon in the trigger toolbar.

4. Go to the **Web Service** tab.
5. Define the properties of the Web Service and port.
6. Go to the **Action** tab.
7. Define the actions that should be executed, when the trigger occurs.
8. Click on the **OK** button.
   The trigger is ready for use.

If the Web Service trigger provides data values for the labels, you will have to extract the values from the incoming data and transfer them to the variables. Do the following:

1. Double click the Web service trigger to open its properties.
2. Go to the **Variables** tab.
3. Define the variable that exist on the label.

Note:
You can manually define the variables or import them from the label file. If creating variables manually, make sure to use the same names as the names defined in the label.

4. Go to the **Filter** tab.
5. Tick the option **Use filter** to enable filter functionality.
6. Select the appropriate **Filter type** that matches the format of the incoming data.
7. Define the fields in the incoming data stream.
8. Link the fields with the variables.

9. Click on the button **Verify the Definition** to test the processing of your filter.
10. Click on the **OK** button.

Using SAP AII Trigger

What is the SAP AII Trigger

The SAP AII trigger detects incoming information from SAP Auto-ID Infrastructure through the HTTP protocol. The connection through the AII infrastructure is essentially identical to a RFC connection (for the purposes of NiceWatch and labeling), but it includes a predefined format of the XML output file.

In order for NiceWatch to receive data from SAP ERP, the RFC destination must be set up in SAP ERP (transaction SM59).

Note: This functionality is only available with the NiceWatch Enterprise Business Connector product.

Using the SAP AII Trigger

Set a RFC destination (transaction SM59) - connection type G

In SAP ERP, on the **Technical settings** tab, make the following entries:

- In the **Target host** field, enter the IP of the NiceWatch Server.
- In the **Service No.**field, enter the port where NiceWatch Server will be waiting for incoming messages from SAP. You will need to set the same port in NiceWatch configuration.

Under Logon/Security, setup the connection type. The NiceWatch device controller supports the following: o No logon o Basic Authentication

Set the trigger for receiving messages from SAP ERP:

- In the Edit menu, select Add trigger and then select SAP AII (or click on the SAP AII trigger icon in the Triggers toolbar.
- In the Trigger Properties window, make the following configuration:
    a. **Trigger Name:** select the name of your trigger
    b. **Port:** select the port where SAP is going to send the messages. This port needs to correspond to the *Service No.* field in the *Technical settings of the RFC destination in SAP* (see RFC destination settings).
    c. Press the **OK** button to save the settings and start the trigger.

Note: If you wish to secure the data exchange with a password, check the **Basic authentication** checkbox and enter the username and password. This username and password must match the one set in SAP RFC destination settings.

To test the communication, go to SAP RFC destination settings and press the **Test connection** button. If all settings are configured correctly:

- SAP will display a "*Status http response 200*"
- NiceWatch will display the "*Trigger was executed*" message.

Using the SAP AII Trigger Input in Labels

NiceWatch enables you to send data in various structured types. The most commonly used formats are .csv and XML. Other types of data sources are also available, but are not recommended as they require more complex setup.

The AII infrastructure features a pre-defined XML format, so there is no need to make filter settings or connect variable values. The variables are predefined to match the fields provided in the AII XML, and are ready to be used on labels.

Note: You can send the data from SAP to the NiceWatch RFC destination. NiceWatch will save the data to your hard disk or network destination if you select the Save incoming data to the file checkbox on the SAP AII trigger properties tab.

As the AII trigger gathers all needed information from the XML file, there is no need to set actions to print a label. The actions editor is available primarily as a tool to append other actions to the data gathering and printing operation.

Using HTTP Trigger

What is the HTTP Trigger

The HTTP Trigger detects incoming information from data sources through the HTTP protocol.

In order for NiceWatch to receive data from SAP ERP, the RFC destination must be set up in SAP ERP (transaction SM59).

Note: This functionality is only available with the NiceWatch Enterprise Business Connector product.

Using the HTTP Trigger in SAP

Set a RFC destination (transaction SM59) - connection type G

In SAP ERP, on the **Technical settings** tab, make the following entries:

- In the **Target host** field, enter the IP of the NiceWatch Server.

- In the **Service No.**field, enter the port where NiceWatch Server will be waiting for incoming messages from SAP. You will need to set the same port in NiceWatch configuration.

Under Logon/Security, setup the connection type. The NiceWatch device controller supports the following: o No logon o Basic Authentication

Set the trigger for receiving messages from SAP ERP:

- In the Edit menu, select Add trigger and then select HTTP (or click on the HTTP trigger icon in the Triggers toolbar.
- In the Trigger Properties window, make the following configuration:
  a. **Trigger Name:** select the name of your trigger
  b. **Port:** select the port where SAP is going to send the messages. This port needs to correspond to the *Service No.* field in the *Technical settings of the RFC destination in SAP* (see RFC destination settings).
  c. Press the **OK** button to save the settings and start the trigger.

Note: If you wish to secure the data exchange with a password, check the **Basic authentication** checkbox and enter the username and password. This username and password must match the one set in HTTP destination settings.

To test the communication, go to SAP RFC destination settings and press the **Test connection** button. If all settings are configured correctly:

- SAP will display a "*Status http response 200*"
- NiceWatch will display the "*Trigger was executed*" message.

Using the HTTP Trigger Input in Labels

NiceWatch enables you to send data in various structured types. The most commonly used formats are .csv and XML. Other types of data sources are also available, but are not recommended as they require more complex setup.

If you already have a sample data source (csv or XML file) that will be sent by your data source, you can very easily import the structure into NiceWatch.

Note: You can use this trigger to send the data from SAP to the NiceWatch RFC destination. NiceWatch will save the data to your hard disk or network destination if you select the Save incoming data to the file checkbox on the HTTP trigger properties tab.

1. Save the data source file saved on the file system.
2. When you have access to the data, select the **Filter** tab in the **HTTP Trigger** properties window.
3. Check the **Use filter** checkbox.

4. From the **Filter type** dropdown, select the type of your data source.

If your data source is a .csv file

1. Select the **Text Database analysis**
2. Press the **Get fields** button and point to your sample csv file to import the structure.

If your data source is an XML file

1. Select the **XML database analysis**
2. Press the **Get the structure** button and point to your sample XML file to import the structure.

After importing the structure, connect the structure to the variable fields in the label template. If the variable field names in the .lbl file match the ones in your data source, NiceWatch will connect them automatically.

Defining Actions

Overview of Actions

Every event must have a defined action. When an event occurs (file drop, serial port communication, email or TCP/IP communication) the trigger actions tell the middleware integration server what to do. Actions include commands like the following:

- "open the label"
- "set the variable on the label to some value"
- "select printer for printing"
- "print the required amount of labels"

If an action cannot be executed, the application records an error description in the log file that helps to identify and resolve the problem.

Advanced Options for Load Variable Data

Advanced options for the text file containing the exported data are defined in this dialog box.

Variables: Select which variables you want to load from the text file.

- **All variables: The default setting is to load all variables.**
- **Selected variables: Click on the Select button and select the variables you want to load.**

Information of the Text File Structure: Define the delimiter and text qualifier as you want to use in the text file. Make sure you select the same parameters that were used in the action Save variable data.

Other: Select the proper data encoding type from the list of options. If you are in doubt what to use, leave the setting to the default Automatic detection and the application will check for the encoding.

Advanced Options for Save Variable Data

Here you can define the advanced options for text file that will contain the saved data.

Variables: Select which variables you want to save to the text file.

- **All variables: The default setting is to load all variables.**
- **Selected variables: Click on the Select button and select the variables you want to load.**

If File Exists: Define the action, if the file already exists. You can always create a new file, or append data to the existing file.

Note: If you will create a new file, any existing file with the same name will be overwritten. You might loose some data, so be careful when selecting this option.

Information of the Text File Structure: Define the delimiter and text qualifier as you want to use in the text file. Make sure you select the same parameters later when using the action Load variable data.

Other: Select the proper data encoding type from the list of options. If you are in doubt what to use, leave the setting to the default Automatic detection and the application will check for the encoding.

Open Document/Program

The action executes the specified program or opens the specified document, when the Trigger is executed.

File Name: Define a file name for the document/program that you want to open. Include full path to the program or document.

Note:
If you are opening a program, you can also include variables as parameters in the command line. Enclose the variables in the square brackets. For example: c:\program files\my_program.exe [Variable]

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Open Label

This action defines the label file that will be used to printing, when the trigger is executed.

A good practice is to provide a full path and filename to the label file, not just the label name alone. If you provide the label name alone, the application will look up the user's default label folder as defined in application preferences. When running the application in service mode as SYSTEM account, provide the UNC path to the file, not mapped drive letter.

Label: Define the fixed name of the label or select the variable containing the path and name to the label.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Note: The action Open Label can contain other actions nested below it. For example, some actions cannot exist on their own, but are always nested below Open Label action, such as Print label, Set Printer or Close label.

Close Label

Use this command to force closing the label that you previously opened with a command Open Label.

Do not use this action if printing speed is important to you. By default,NiceWatch will open the label and keep it open as long the application runs. Keeping the label open saves you time you would otherwise spend for re-loading the label. In time-critical label-printing environments every second can count.

Note: NiceWatchwill not see the changes in the label file that label designer made and will continue to print the older label contents. If the design of your label templates changes frequently, make sure to use the action Close Label, otherwise don't use it to speed up printing.

Print Label

This action executes the printing command and defines how many labels will be printed.

Number of Labels: Define how many labels should be printed.

- **Fixed:** The number entered in the edit field defines how many labels will be printed.
- **Unlimited:** The maximum quantity of labels supported by the current printer will be printed. This options is generally used with database files, where unlimited would be used as: print the whole database.

Note:
Be careful and do not create an infinite loop, when label printing would not end.

- **Variable quantity:** Some variable on the label will set the quantity of labels to be printed. In most cases this will be some field from the database. The application does not know how many labels will be printed. When the data is processed on the label, the variable quantity gets its value.
- **From variable:** The value of the selected variable defines the quantity of labels that will be printed.

Advanced Options: Click this button to set advanced options for the quantity of labels.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Advanced Options for Label Quantity

Number of Skipped Labels: Defines the number of labels that will not be printed. The option is only applicable for the first sheet of labels, not for each sheet. The option is useful if some labels on the sheet have already been printed and you do not want to print them again.

Note:
The option for skipping labels is applicable when printing labels to office printers. It is not available when using thermal printers.

Identical Label Copies: The option sets the number of label copies you want to use with every printed label.

Label Sets: The option instructs how many times the printing process should be repeated.

For example:
If you set Number of Labels to Fixed 5 and Number of labels sets to 3 in the Advanced Options, the printing of 5 labels will be repeated three times, resulting in 15 printed labels.

Run Command File

Executes the commands in the specified command file. The following types of the command files are supported:

- JOB file
- XML file
- CSV (Comma Separated Values) file

All of the command files provide the commands to instruct the print engine what to do. In most cases you want to open the label, set the values to variables on the label and print

the label to the specified printer. See the chapters in the section **Automating the Software**.

File Name: You can set a fixed command file name or let some variable define it.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Send Custom Commands

Send the defined list of NiceCommands to the print engine for execution.

Commands: Enter the NiceCommands in the text area.

Edit: Click on the Edit button to open the Expression Editor that will help you define the script of custom commands.

You can also include variables in the commands. Just type in the variable name and enclose it with the square brackets.

For example:
To print the number of labels as specified in the variable Quantity, the print command should look like PRINT [Quantity]

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Note:
For more information about NiceCommands refer to the help or User Guide.

Load Variable Data

This action reads the data from the text file and sets values to the variables on the form.

File Name: Specify the name of the text file with data. It can be fixed or variable.

Advanced Options: Click on this button to select variables you want to load from the text file and define the structure of the text file.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Save Variable Data

This action saves the data from the variables to the text file.

File Name: Specify the name of the text file with data. It can be fixed or variable.

Advanced Options: Click on this button to select variables you want to load from the text file and define the structure of the text file.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

For Loop

For loop makes it possible for the action to be executed a certain number of times.

Loop from: Select a starting value for the incremented loop.

Loop to: Select an end value for the incremented loop.

Note:
If the loop from value is higher than the loop to value, the loop will decrement the value regressively.

The **Loop from** and **Loop to** values can be fixed or variable, and the current loop value can also be saved into a variable.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Select Variables

Select the variables which values you want to save/load to/from file. Use Up and Down arrows to set the order of appearance in which values are defined in the text file.

Set Printer

The action defines the printer name that you want to use for label printing. This setting will override the printer setting that were set directly on the label.

Note:
This action is useful, if you want to print one label file to several different printers without the need for creating duplicated label files linked to different printers.

Printers: Define fixed name of the printer or select the variable name containing the name of the printer.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Set Variable

The actions sets the defined value to the selected variable.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Run Oracle XML

Use this command to print labels based on the Oracle XML file. You do not have to manually define the XML database analysis filter, just point to the Oracle XML file, and the application will do the rest.

When you need to print the labels from Oracle application, the application generates an XML file and encodes the following information into it:

- label filename
- printer name
- quantity of labels to print
- data for variables on the label

This action can open the Oracle XML file, extract the values from it, open the correct label and print it in specified quantity to the specified printer.

Note:
Labels from Oracle XML files can also be printed using the **XML Database Analysis** filter. You need to use the filter, when you want to use the values from XML file in some other actions. If you only want to print the labels with data from the XML file, just use the action **Run Oracle XML**.

A typical Oracle XML file might look like this:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE labels SYSTEM "label.dtd">
<labels _FORMAT="label.lbl" _JOBNAME="SampleJob001" _QUANTITY="1"
_PRINTERNAME="Sample Printer">
  <label>
    <variable name="item">WS007</variable>
    <variable name="lot">A12345</variable>
    <variable name="serial">92837</variable>
    <variable name="item_desc">Weight scale</variable>
  </label>
</labels>
```

File Name: You can set a fixed Oracle XML filename or let some variable define it.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Visual Basic Script

This action defines the programming script that you can use for advanced data manipulation.

The action has the possibility to use all standard VB Script functionality. In addition the script is also able to get and set values of the variables and to read and write to/from the serial port.

Visual Basic Script: Define the script in the text area.

Build Script: Click on this button to open the Expression Editor that will help you construct your script.

Check Script: Click on this button to verify the syntax of you script. If there is a syntax error in the your script, you will be notified about it.

Refresh Database

The Refresh Database action refreshes all elements that are connected to a database, to reflect changes made elsewere.

Write Data to Serial Port

Define how the data will be written to the serial port.

The data sent to the serial port is stored as a fixed value or in the generated variable. You can use the value of the variable directly on the label or use it in some other action in the form (like Visual Basic script if additional data-manipulation is required).

Serial Port Settings: Define the properties of the serial port where you have attached the serial device. The parameters in this dialog box have to match the settings on the device. Refer to the documentation of your serial communication device so you will be able to specify the correct communication parameters. If the settings on the device and in the software do not match, communication will not be possible.

Data: Set the data to be sent to the serial port as fixed data or data from a variable. Type in the set of characters that must be sent to the device. Click on the small arrow button on the right to display list of all available special characters.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Read Data from Serial Port

Define how the data will be read from the serial port.

The data received on the serial port is stored in the generated variable. You can use the value of the variable directly on the label or use it in some other action in the form (like Visual Basic script if additional data-manipulation is required).

Serial Port Settings: Define the properties of the serial port where you have attached the serial device. The parameters in this dialog box have to match the settings on the device. Refer to the documentation of your serial communication device so you will be able to specify the correct communication parameters. If the settings on the device and in the software do not match, communication will not be possible.

Initialization Data: Initialization data is set of characters that need to be sent to the serial device in order to activate it and make it ready to send data to the PC computer. Type in the set of characters that must be sent to the device. Click on the small arrow button on the right to display list of all available special characters.

> **Send initialization data:** Tick the option to enable initialization data. This is optional feature.
>
> **Wait before starting to read data:** You can define the time-out in milliseconds after which the receiving data will begin. This is optional feature.

Filter: You can filter the data you receive from the serial port and cut only the important characters. You can limit what data you will receive. This is optional feature.

> **Use start and stop string:** Define the two strings that will enclose the data you need to extract. For example, if start string is **###** and stop string is **$$$**, the extracted data from the raw data **q###12345$$$1** is **12345**.
>
> **Use start and end position:** Define the start and stop position of characters you want to extract. For example, if start position is **10** and stop position is **14**, the extracted data from the raw data **q###12345$$$1** is **12345**.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Verify License

This action allows the user to insert a license check to the trigger.

- If the entered **Solution ID** number is not the same as the ID of the application, no actions will be executed.
- If the entered value is 0, actions will be executed if any valid license is found.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term

is met. The action will be started only if condition returns True as the result of the expression.

Printer Status Report

The action allows you to obtain the information about the current printer status.

Printers: Select the printer for which you want to be notified about the status change. You can select a printer from the list of the locally available printers, or get the printer name from the selected variable.

Rise action if printer is in error state: Enable this option to be notified each time the printer enters the error state (out of ribbon, out of paper, out of labels, not accessible, etc.)

Rise action if printer status is OK: Enable this option to be notified each time the printer exists the error state. The printer becomes accessible again.

Report destination: Define the destination where you want to publish the message. For TCP/IP trigger you can respond back to the 3rd party application that provided the label data (or you can send the message to some other server). For all other trigger types you can save the message to text file and store it to local or network disk.

Report contents: Type in the message you want to receive each time the printer changes its state and you have defined to be notified about that change. Click on the **Edit** button to use more powerful text editor. Inside the editor you can also embed some internal variables into your message. The variables will provide more information about the print job that caused the change in the printer state and more detailed messages. The available internal variables are: *SpoolerStatusID*, *SpoolerStatus* and *PrinterStatus*.

Report encoding: Define the encoding type you want to use for the report message.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Note: The action is available in NiceWatch Enterprise edition only.

Web Service

The Web Service action browses for a Web Service WSDL document. Based on the definition, the user can select operations to be used as parameters (fixed or variable), which return values depending on a *success* or *fail* condition.

WSDL Importer: The WSDL importer enables you to import a WSDL file, containing Web Service operations.

Operation: This is the operation that will be called on the Web Service. The operation is retrieved from a WSDL import and displayed here including with its type and value type.

Condition: This is a Visual Basic Script boolean expression. Only two results are possible (True and False). Use this option to enable current action only when some term is met. The action will be started only if condition returns True as the result of the expression.

Note: This functionality is only available with the NiceWatch Enterprise Business Connector product.

Using Filters

Working with Filters

The middleware integration server includes powerful filter capabilities to parse input data, extract required data fields regarding your criteria and map them to the variables defined in the label. If the data you have received from your trigger (file, COM, email, TCP/IP, database, Web service) cannot be used immediately, you have to use filter options to extract only the portions of the data that you need to print the labels. Mapping data filters is an optional function that you only need to set up if the data from your trigger cannot be used immediately to print the label.

The application offers three filter types to process data:

- Document analysis of unstructured text files
- Text database analysis of structured text and database files
- XML database analysis of XML files

Note:
The data for filters is provided from your trigger. The data received by the trigger will be used as the input data for the filter.

Selecting the Filter Type

Selection of the proper filter type for your trigger is based on the structure of the input data. The received data is usually formatted in one of the following formats:

unstructured text files	**Document analysis filter** **This filter can be used with data that has no structure whatsoever. The perfect example for this filter is a report file or printer data stream file (text file with printer commands). Use this filter to define data-parts within received data, that you want to assign to certain variables.**
structured text files	**Text database analysis** **This filter can be used with a classic text database files that hold the data for variable fields on the label. Data**

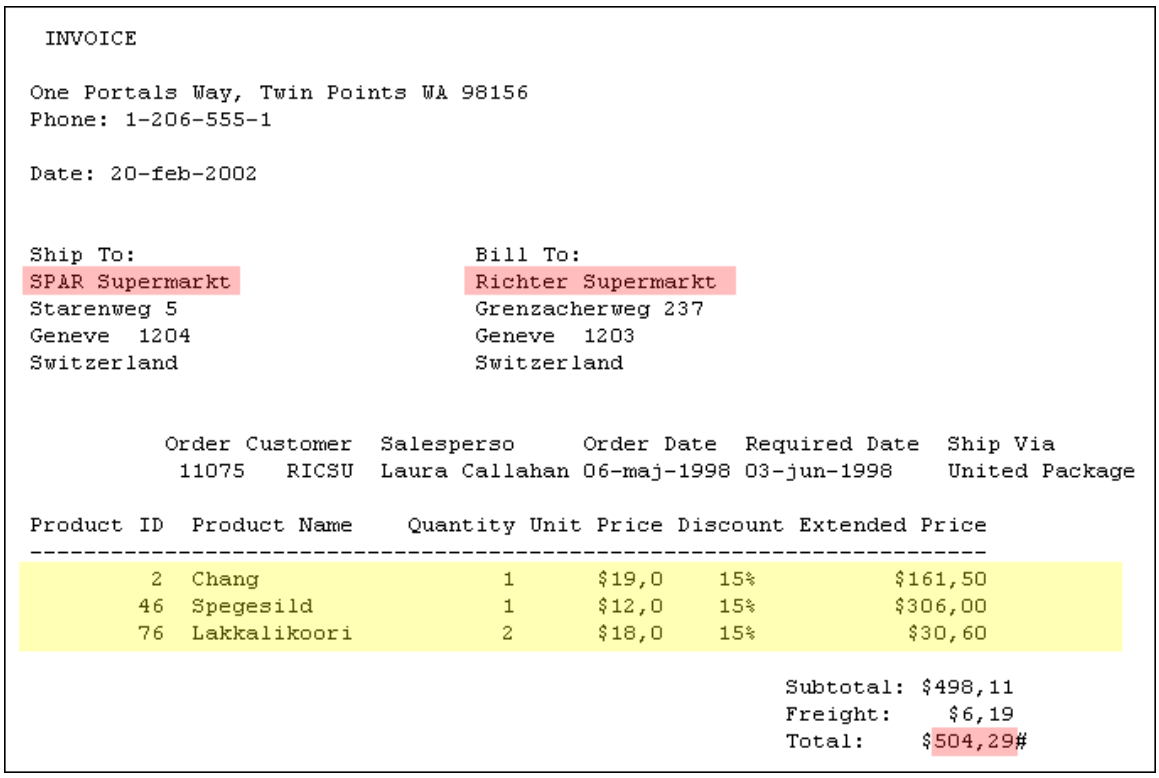
	can be encoded with: ▪ **Fields of fixed width or** ▪ **Fields, delimited with some character. Different delimiters can be used for separators between field values.**
XML files	**XML database analysis** **This filter can be used with data formatted in XML files.**

Filter Document Analysis

Overview of Document Analysis Filter

The filter Document Analysis parses an unstructured text file for data that you want to print on your label. Unstructured text files are for example reports, invoices or printer data stream files. ‘Unstructured’ refers to unformatted data structure with data not organized in a table-like format. The data is not organized in rows and columns that are separated with a character like a comma or have fixed column width. The application can extract specific data items within an unstructured file that you want to assign to certain variables.

The invoice below shows the highlighted data that you want to print on your label. Usually, you extract the lines with the data and then use the data fields individually on the label.



```
 INVOICE

One Portals Way, Twin Points WA 98156
Phone: 1-206-555-1

Date: 20-feb-2002


Ship To:                        Bill To:
SPAR Supermarkt                 Richter Supermarkt
Starenweg 5                     Grenzacherweg 237
Geneve  1204                    Geneve  1203
Switzerland                     Switzerland


          Order Customer  Salesperso     Order Date  Required Date  Ship Via
           11075   RICSU  Laura Callahan 06-maj-1998 03-jun-1998    United Package

Product ID  Product Name    Quantity Unit Price Discount Extended Price
-------------------------------------------------------------------------
         2  Chang                  1      $19,0    15%          $161,50
        46  Spegesild              1      $12,0    15%          $306,00
        76  Lakkalikoori           2      $18,0    15%           $30,60

                                                  Subtotal: $498,11
                                                  Freight:    $6,19
                                                  Total:    $504,29#
```

Unstructured file: Data Text File

The column "Quantity" in the invoice determines the amount of labels that will trigger to print (a total of four (4) labels).

The figure below shows another type of unstructured document (printer stream file) that you parse with the Document analysis filter. The application extracts the highlighted data fields and maps them to the variables on the label. Data fields from one printer's data stream can be printed on some other printer.

```
^XA~TA000~JSO^LTO^MMT^MNW^MTT^PON^PMN^LH0,0^JMA^PR6,6^MD0^JUS^LRN^CI0^XZ
^XA^LL0480
^PW719
^FT399,171^A0N,28,28^FH\^FDRichter Supermarkt^FS
^FT52,171^A0N,28,28^FH\^FDSPAR Supermarkt^FS
^FT294,231^A0N,28,28^FH\^FD2^FS
^FT294,268^A0N,28,28^FH\^FDChang^FS
^FT294,305^A0N,28,28^FH\^FD1^FS
^FT295,380^A0N,28,28^FH\^FD€349,10^FS
^FT294,343^A0N,28,28^FH\^FD€190,00^FS
^FT152,89^A0N,68,67^FH\^FDReport Label^FS
^FT397,140^A0N,28,28^FH\^FDBill To:^FS
^FT52,140^A0N,28,28^FH\^FDShip To:^FS
^FT52,231^A0N,28,28^FH\^FDID:^FS
^FT52,268^A0N,28,28^FH\^FDProduct name:^FS
^FT52,305^A0N,28,28^FH\^FDQuantity^FS
^FT94,448^A0N,23,24^FH\^FDSample Printed from Label Software.^FS
^FT52,343^A0N,28,28^FH\^FDPrice:^FS
^FT52,380^A0N,28,28^FH\^FDTotal:^FS
^PQ10,0,1,Y^XZ
```

Unstructured file: Print Stream Data File

The parsing and mapping filter technology works with any kind of unstructured file that you want to use as a source of variable data on your labels. The filter allows you to easily automate the extraction of the required data fields and print them on the label.

Mapping the Document Variables

The dialog box Document Field Properties is used to link the fields from the document to the variables. The dialog box is also used to define the position of the fields in the document. The same dialog box is used for both, document fields and fields from the repeatable area.

Note:
The difference between the document fields and fields from the repeatable area is that document fields appear absolutely on the document, while repeatable fields appear relatively in the repeatable area. You have to take this into consideration when defining the field start/stop positions.

If you are using only document fields, the input data from trigger will be used on all labels. If you are using repeatable variables, each item in the repeatable area will be used on one label. If the repeatable area contains 10 rows, the data from each row will be used on one label. You can combine the document and repeatable variables on the label. Document variables will stay the same for all occurrences of repeatable variables and will be the same for all printed labels for the trigger, but repeatable fields will change for each item.

Note:
You can use the data from more consecutive lines on a single label. Change the option **Number of rows of the repeatable part** when defining the repeatable area.

To create the document fields or fields in the repeatable part and link them to the variables, do the following:

1. Click on the Add button to open **Document Field Properties** dialog box.
2. Define the field name.
3. Select the variable that will be linked with the field. The value of the field will be transferred to the variable when the filter is processed.
4. Define the parameters for the **Field Start** and **Field Stop**.

Note:
The start and end of the area is defined using three parameters. Row number, Column number and Prefix word. Only two parameters are required to successfully define the start/stop area. For example, if you know that the data will start in row number 10, and that required data is always preceded with string "Data #", you would enter 10 to **Row number** "Data #" to **Prefix word** and leave *Column number* to 0.

5. Tick the option **Truncate spaces before the first and after the last character**, if you want to remove the trailing and leading spaces from the value.
6. Repeat this procedure from step 1 for all other fields you want to set.

Note:
If no variables are defined, go to the Variable tab and define the variables.

Filter Text Database Analysis

Overview of Text Analysis Filter

Parses a structured text file for data that you want to print on your label. Structured text files are ASCII text files with label data, text files with label data and optional label and printer information, or XML files. ‘Structured’ refers to formatted data structure with data organized in a table-like format, using a fixed width of data fields or data fields separated with a certain character.

The samples below show a text database with fixed length of the fields that are formatted in columns and a text database using commas to separate the data fields. Each line in the file contains a record with data fields for a single label. Each column provides data for one variable on the label. If the trigger data is available in a structured form, the application can use the data immediately for variable fields on the label without performing additional configuration steps.

```
1161890-03 1161890 378 67871130 HALTER
1162008-03 1162008 884 67871350 FUSSRASTENPL
1162132-03 1162132 150 67871125 BREMSHEBEL
1162212-03 1162212 105 67871134 LAGERBOCK
```

Structured file: ASCII Text Database

```
Test,200,2
Software,101,1
Printing,55,3
```

Structured file: Comma-separated Data Fields

Mapping the Text Database Variables

To link the text database fields to the variables, do the following:

1. Define the structure of your text database. All fields must be visible in the text are in **Text Database Analysis Option** section.
2. Double click on the field entry in the list.
   The dialog box **Field Properties** will open.
3. Select the variable that will be linked with the field. The value of the field will be transferred to the variable when the filter is processed.
4. Tick the option **Truncate spaces before the first and after the last character**, if you want to remove the trailing and leading spaces from the value.
5. Repeat this procedure from step 2 for all other fields you want to set.

Note:
If no variables are defined, go to the Variable tab and define the variables.

Filter XML Database Analysis

Overview of XML Database Analysis

The filter XML database analysis parses an XML file for data that you want to print on your label. The sample below shows the data from the invoice as an XML file output from Oracle. The sample is similar to the invoice sample from **Document Analysis Filter**, but it is formatted as XML file, not plain text file.

The application parses the file and triggers printing the values on a label. The XML structure provides the elements, attributes and the name of the variables that are printed on the label.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE labels SYSTEM "label.dtd">
<labels _FORMAT="C:\My Documents\My Labels\xmlinvoice.lbl"
_JOBNAME="SampleJob001" _QUANTITY="2" _PRINTERNAME="ZEBRA R4MPlus">
  <ShipTo>SPAR Supermarkt</ShipTo>
  <BillTo>Richter Supermarkt</BillTo>
  <label>
    <variable name="ID">2</variable>
    <variable name="ProdName">Chang</variable>
    <variable name="Quantity">1</variable>
    <variable name="Price">€190,00</variable>
  </label>
  <label>
    <variable name="ID">46</variable>
    <variable name="ProdName">Spegesild</variable>
    <variable name="Quantity">1</variable>
    <variable name="Price">€120,00</variable>
  </label>
  <label>
    <variable name="ID">76</variable>
    <variable name="ProdName">Lakkalikoori</variable>
    <variable name="Quantity">2</variable>
    <variable name="Price">€18,00</variable>
  </label>
  <TotalPrice>€349,10</TotalPrice>
</labels>
```

Structured file: Oracle XML File

Mapping the XML Fields to the Variables

Once you have the structure defined, you must map the XML data types to the variables.

Do the following:

1. Define the structure of your XML database. All fields must be visible in the text are in **XML Database Analysis Option** section.
2. Double click on the field entry in the list.
   The dialog box **Field Properties** will open.
3. Tick the variable that will be linked with the field. The value of the field will be transferred to the variable when the filter is processed.
4. The data type that needs to provide recurring data needs to be defined as recurring element. Only these data types can be marks as repeatable block, a property that ensures the recurring elements.
5. Tick the option **Truncate spaces before the first and after the last character**, if you want to remove the trailing and leading spaces from the value.
6. Repeat this procedure from step 2 for all other fields you want to set.

Note:
If no variables are defined, go to the Variable tab and define the variables.

Using Serial Port Communication

Defining Data Polling

This option data polling enables constant communication with the serial device. In specified time intervals the defined data characters are sent to the device. The device understands the sequence of characters as command to send the data to the PC computer.

There are two methods of defining the data for polling:

1. The characters you want to send to the device can be entered in the memo box below. Use the button with arrow on the right to enter special characters.
2. If your serial device can return the information of its status, you can use the built-in Visual Basic functionality and write a script that will communicate with the device. Visual Basic Script functionality lets you manipulate the incoming data. Two functions are pre-defined and must existing in your script. DataAvailable and ReceiveData both return Boolean values.

DataAvailable	**This function is used to acquire data from the serial device and validates the incoming data for consistency.** **You can use internal functions Comport.Send and Comport.Receive to communicate with the device. Learn more about them in the topic** Defining Initialization Data**.** **If you set the function to return 1, the ReceiveData function is then executed. If you set it to return 0, no incoming data is available and action is not triggered.** **The focus is returned to the next polling interval.**
ReceiveData	**You can pass the incoming data from the serial device acquired in DataAvailable function to this function. ReceiveData function can then be used to manipulate incoming stream of data to suit your needs. You can re-arange data fields, add or remove text and make any other data operations.** **If you set the function to return 1, the trigger action is then started. If you set it to return 0, it will be a signal for the trigger not to start the actions.** **The focus is returned to the next polling interval.**
ClearBuffer	**This command clears input and output buffers of the serial port on the PC computer.**

Defining Initialization Data

Initialization data is set of characters that need to be sent to the serial device in order to activate it and make it ready to send data to the PC computer. There are two methods how the initialization data is defined:

1. If you have a dummy serial device that is activated by a certain combination of characters, you can type in such characters. Click on the small arrow button on the right to display list of all available special characters.
2. If your serial device can report its current working state back to the computer, you can use Visual Basic script and communicate with the serial device. You can program your script code to handle bidirectional communication to the serial device.

When you are using bidirectional communication, the ordinary scenario is to send some initialization string to the device and then listen for a reply. If the reply is what you are expecting it to be, the initialization was successful. If the reply is not valid, the initialization failed, an error in logged the system log, but trigger will still be checked.

Two special functions are available in Visual Basic script for data sending and receiving to/from serial port.

Comport.Send	**ComPort.Send(Data: string, Length: integer): integer** **The ComPort class function Send has two parameters. The first parameter contains the data that will be sent to the serial port. The second parameter must contain the length of the sent data. Function return integer values:** **1 - if writing to serial port was successful and** **0 - if error occurs.**
Comport.Receive	**ComPort.Receive(TimeOut: integer): string** **The ComPort class function Receive has parameter TimeOut. This parameter contains the time (in seconds), how long the function will wait for the incoming data.** **The function returns data if it is available in the specified time frame. If the data is not received, the function returns empty string.**

The built-in function InitData must be part of the script code, but you are free to modify it. Use the functions Comport.Send and Comport.Receive to exchange data with the serial device and then decide if the initialization was successful or not.

Then set the output of the InitData function accordingly to let the application know the result of the initialization sequence.

Note:
The initialization string is sent to the serial device only at a time when you start trigger checking in the application or when you restart the service.

Using Toolbars

Using Server Toolbar

	**Click on this button to start the middleware integration server.** **Note:** **If middleware server is not running, no triggers events are detected and no actions are started.**
	**Click on this button to stop the middleware integration server.**
	**Refresh the connection to the server. The configuration of triggers, trigger status and server parameters are obtained from the server.**

If you do not see the server toolbar, do the following:

1. Select the command **Toolbars** from the View menu.
2. Enable the **Server** toolbar.

Using Standard Toolbar

	**Create a new .MIS (Middle-ware Integration Server) configuration file. The configuration file stores the properties of the trigger events, filters and actions you define.**
	**Open settings from an existing configuration file.**
	**Save the existing settings into the configuration file.**
	**Copy the definition of the selected trigger to the clipboard.**
	**Paste the configuration of the trigger from the clipboard.**

If you do not see the standard toolbar, do the following:

1. Select the command **Toolbars** from the View menu.
2. Enable the **Standard** toolbar.

Using Trigger Toolbar

	**Start monitoring trigger events.**
	**Stop monitoring trigger events.**
	**Define new trigger using Trigger Wizard.**
	**Define new file trigger.**
	**Define new email trigger.**
	**Define new COM port trigger.**
	**Define new TCP/IP trigger.**
	**Edit the selected trigger.**
	**Delete the selected trigger.**

If you do not see the trigger toolbar, do the following:

1. Select the command **Toolbars** from the View menu.
2. Enable the **Trigger** toolbar.

Working with Variables

Using Variables

Generally you use variables to store some data value. The source of data can be some filter or some action. Once a variable has a value, you will generally use it to assign it to the object on the label.

To assign the value to the variable on the label just make sure the names of the variables (defined in the trigger and defined in the label) are the same. You can define the variables in the trigger manually, but it is recommended to simply import the variables from the label.

Variables can either be numerical, alphanumerical, or values including symbols, both in the form of condensed statements and multiple lines.

You can use variables for different purposes:

- You can use them in filters to store values from data mappings.
- You can use them in actions, like Set Variable, Visual Basic Script, Open Label (to open label with variable name), Set Printer (to print labels to printer with variable name), Print (to print variable quantity of labels), etc.
- You can use them in conditions. You can define a condition for each action. The action runs only when a condition is true.
- You can use them to exchange data among applications with the actions Load and Save variable data.
- Etc.

Editing Variables

You can only edit internal variables. Internal variable is a variable that you manually created in the trigger options.

If a variable is defined on the label and you imported it from the label file, you cannot make any modifications to it. When you want to change such imported variable, you will have to make modifications to the variable on the label and import it again.

Types of Variables

Each type of variable has its own icon to easier distinguish variables among each other. The icon in front of the variable name determines its type.

The available types of variables are:

	**Prompt variables that are defined in a label file.** **The application reads these variables from the label file and show them in the list.**
	**Database variables that are defined in a label file.** **The application will read these variables from the label file and show them in the list.**
	**Local variables.** **These variables are local to the for and are not included in the label file. You can use these variables in the actions.**
	**Internal variables.** **These variables are local and are not included in the label file. You can use these variables in the actions. Internal variables cannot be modified, you can only use them.**

Internal Variables

Internal variables are filled automatically by the software and you do not have any influence on them. They cannot be edited and modified, but only used in the application. Their value is updated for every printed label.

Internal variables are represented with the  icon so they can be easily distinguished from the other types of variables.

The list of available internal variables:

Hostname	**The IP address of the connecting hosts, if using TCP/IP trigger.**
ComputerName	**Contains the information about the computer name of a PC computer where labels are processed, as specified from Windows system.**
DataFileName	**Contains full name of the temporary file with data, including the path to the file (c:\trigger\file.dat).** **This is a temporary file that contains input data received from any type of trigger (file, email, serial port, TCP/IP) and is actually processed in the program.**
Date	**Contains the current system date.**
DefaultPrinterName	**Contains the name of the default printer on the system.**
PathDataFileName	**Contains the path to the temporary file with data (c:\trigger\).**
ShortDataFileName	**The same as DataFileName but contains only the filename without the path.**
ShortTriggerFileName	**Contains the name of the trigger file that actually started the actions. Only has a value with file triggers.**
SystemUserName	**Contains the name of the system Windows user that is logged in and is running the application.**
Time	**Contains the current system time.**
TriggerFileName	**The same as ShortTriggerFileName but contains also the full path to the file**
UserName	**Contains the name of the user as defined in the labeling software. Only has a value if you have enabled user management.**

Manual Creation of a Variable

To manually create a variable, do the following:

1. Open the properties of the trigger.
   If you already have a trigger defined, you can just double click it. Otherwise create a new trigger.
2. Go to the **Variables** tab.
3. Click on **Add** button. The *Variable Properties* dialog box will open.
4. Enter the variable properties then click **OK**.
5. Click **OK** to close the trigger properties.

Variable Extraction from the Label

To extract the variables from the label, do the following:

1. Open the properties of the trigger.
   If you already have a trigger defined, you can just double click it. Otherwise create a new trigger.
2. Go to the **Variables** tab.
3. Click on **Get Variables** button. The *Select Label Files* dialog box will open.
4. Browse to the label file from where you want to import the variables. Only prompted variables will be imported into the trigger. The import procedure will ignore any other type of variables that might exist in the label.

Note: When importing variables into the label, the link to the label is not remembered. Once you have the variables imported, you can use them to set values to variables on various labels, not necessarily only the label that you used to import the variables. When setting the values of variables on the label the important fact is that you use the same name of the variables.

5. Click **OK** to close the dialog box with list of variables.
6. Click **OK** to close the trigger properties.

Entering Special Characters

The easiest method of entering special characters is clicking on the small arrow button next to the edit fields. The arrow will display a list of special characters. Just click on the character you need.

Alternatively, you can use special characters by using hash sign (#) followed by two characters that represent hexadecimal ASCII code of a character. You can also use caret character (^) to specify escape (ESC, 27) character. To enter # or ^, type it twice one after another (## or ^^).

Limiting Access to the TCP/IP Trigger

To limit connection to your TCP/IP triggers, do the following:

1. Double click your TCP/IP trigger to display its properties.
2. Go to the **TCP/IP** tab.
3. Click on the button **Host Connection Permissions**.
   The dialog box Host Connect Options opens.
4. Type in the IP addresses of the hosts you do not want to connect to your TCP/IP server in the text box **Deny connection from the following hosts**.
   The connection requests coming from the entered hosts will be denied.

Note:
You can also enter the IP addresses from the hosts that you trust. Use the text box **Allow connection from the following hosts**.

Running the Executable in Service Mode

When the middleware integration server executable runs in the service mode, it is started automatically when the Windows operating system loads. There is no need for any of the users to log in and manually start the service.

To start the middleware server in the service mode, do the following:

1. Open the middleware integration client.
2. Make sure the server is not running. To stop the server, select the command **Stop Server** in the Activities menu.

Note:
To stop the server you can also click on the  button in the server toolbar.

3. Select the command **Register Service** from the Tools menu.
   The dialog box Service Preferences will open.
4. For **Startup Type** select how the middleware server should start as service. You probably want to select the option Automatic to be sure the server starts automatically at each restart of the computer.
5. For **Log On As** select under which user the service should run.

Note:
To run the application under domain user type in the user name in the syntax <DOMAIN>\<USER NAME>. To run the application under the local user type in the user name in the syntax .\<USER NAME>. To run the application under a System Account, select the option System Account.

6. Click on the **OK** button.
7. Select the command **Start Server** from the Activities tab.

Note:
To start the server you can also click on the  button in the server toolbar.

When you run the application as a service application and use .JOB files, make sure to use UNC quotation instead of the mapped drives (e.g. LABEL "\\SERVER\MY LABELS\LABEL.LBL" instead of "G:\MY LABELS\LABEL.LBL").

When you run the application under **System Account**, make sure to specify the full path and filenames to the labels you print. Otherwise the application Label will not find the labels. System Account does not have defined any user profile and there are no default folders for labels.

When you run the application under **User Account**, the Windows profile and permissions from the selected user apply. If you specify only label name in the actions, the print engine will search for the label in these default folders.

The best idea is to use the full path and filename to the label you will print from service mode.

Using Data Polling for Serial Devices

To define the initialization data for your serial device, do the following:

1. Select your COM port trigger in the list of triggers.
2. Select the command **Edit trigger** from the Edit menu.

Note:
You can also click on the button  in the trigger toolbar to edit the selected trigger.

3. Go to the **COM Port** tab.
4. Enable the option **Use data polling**.
5. Click on the button **Extract data options**.
6. Define the polling parameters that are required by your serial device.
7. Click on the OK button to close **Extract Data** dialog box.
8. Click on the OK button to close **Trigger Properties** dialog box.

Using Initialization Data for Serial Devices

To define the initialization data for your serial device, do the following:

1. Select your COM port trigger in the list of triggers.
2. Select the command **Edit trigger** from the Edit menu.

Note:
You can also click on the button  in the trigger toolbar to edit the selected trigger.

3. Go to the **COM Port** tab.
4. Enable the option **Use initialization data**.
5. Click on the button **Initialization Data Options**.
6. Define the initialization data that is required by your serial device.
7. Click on the OK button to close **Initialization Data** dialog box.
8. Click on the OK button to close **Trigger Properties** dialog box.

Manually Executing Triggers

Triggers and the trigger-action synergy are the main automation tool offered by NiceWatch. They are designed to supervise a location or communication channel, and trigger actions automatically when an expected change prompts them to do so. However, there is also an option to execute triggers manually.

Right click on the trigger and manual execution options will appear in the bottom of the context menu.

Execute Trigger Now: The trigger is executed with the data defined in the trigger's settings

Execute Trigger with Custom Data: Because trigger input is not available, browse for a data file, which contains the information with which the trigger should be executed. This command is not available for Database triggers, because the information which the trigger needs to process cannot be provided on demand.

Print Preview

NiceWatch enables the user to perform a print preview, which displays the preview of the label print job.

- To access the trigger print preview, right-click the trigger for which you wish to see the preview.
- A dialog window opens, prompting you to select open a file for the preview

The file that is opened before the trigger print preview displays should contain the information that is expected by the trigger. For example, if the preview is selected for an Email Trigger, the file should contain email information which allows the trigger to gather the expected data.

Note: Database Triggers do not prompt you to open a file, because the database used to extract information is already defined in the trigger. The preview is created with the data currently present in the database.

A preview window opens, displaying the labels, as they would be printed if the trigger was executed with the information contained in the file. The information displayed includes:

- **Label selection** - if more than one label is listed in the trigger data file, previews are generated for all labels listed in the left pane
- **Variables** - displays the variables included in the label and the data extracted from the trigger data file for these variables
- **Log** - The NiceWatchlog created for the trigger's execution

Enterprise Print Manager

Introduction to Enterprise Print Manager

Note: The contents of this chapter apply to NiceWatch Enterprise product.

NiceWatch Enterprise is a software package with three modules (NiceLabel Pro, NiceWatch Server, Enterprise Print Manager).

On workstations, users are working with NiceLabel Pro application for label design. As a part of NiceLabel Pro, a service called Label Services is monitoring the local printing process. NiceWatch Server application prints the labels automatically in the background with no user intervention.

Printer and job statuses are sent to the server, where the Enterprise Print Manager (EPM) component is centrally controlling the printing process of the NiceWatch Server. EPM stores all reported client information is into its database (Microsoft SQL Server).

EPM is a web based application. You can use a standard web browser to control it from your machine or from any other workstation in the network. You can view the status of each NiceLabel client, view its event and error logs, manage the print jobs and create alerts.

EPM alerts you upon arrival of a predefined condition, such as an error, a warning, or simple confirmation of executed print job. Several notification options are available: email (through SNMP), Net Send message, RSS 2.0 Feed or SMS message sent to your mobile phone.

You can set up different access permissions to your users. By default full access rights are granted to all users. To change the permissions, refer to the NiceLabel Pro user guide.

To access your EPM, do the following:

1. Open Internet Exporer.
2. Type in the address of your EPM:

   http://*server*/EPM/Home.aspx

Note: The *server* above stands for the name of the computer where you installed EPM.

For more information about how to work with EPM please refer to the EPM user guide.

Automating the Software

Command Files

Use Command Files

You can use command files to instruct the print engine what to do. The command files use the structure of NiceCommands (or equivalent commands) in order to send the commands to the print engine.

The following command files are supported:

- JOB file
- XML file
- CVS file

You can use the command files from several applications:

- From the label designer interactively (command File -> Command Files), or automatically (the name of the command file specified as parameter in the command-line, works for JOB files)
- From form designer in action Run Command File
- From middleware integration module in action Run Command File

The command files are always processed in order from top to bottom of the file. The first command in the file is processed first. The order of appearance is important. For example: you need to specify the printer first and then print the label.

The command files are stored in a text file. Unicode values are supported.

Command File: JOB File

The commands available in the JOB command files are NiceCommands.

See the chapter of NiceCommands for more information about their syntax and method of usage.

Command File: CSV File

The commands available in the CVS command files are a subset from NiceCommands. You can use the following commands: Label, Set, Port, Print and Printer. Of course, the syntax of the commands differs a little bit when used in CVS file.

CVS Command File Structure Explained

The CSV stands for Comma Separated Values. This is the text file where field values are delimited by the comma (,) character. The text file can contain Unicode value (important for multi-language data).

Each line in the CSV file contains the commands for one label printing.

The first row in the CSV command file must contain the column names. This is important for the labeling software to know what is the order of appearance of fields and how is the data organized. Several column names are pre-defined.

Column Name	Description
@Label	**The name of the label to use. It is recommended to include label path and filename. Note: You can provide label name only, but be sure that print engine will try locate the label in the correct folder. Is required.**
@Printer	**Use this field to override the printer defined on the label. Print the label to some other printer. The other printer must be accessible from this computer. Use the printer name for value of this attribute. Not required.**
@Quantity	**Use this field to specify the number of labels to print. Possible values: numeric value, VARIABLE or UNLIMITED. Is required.**
@Skip	**Use this field to specify how many labels to skip at the beginning. This feature is useful if you print sheet of labels to laser printer, but the sheet is partial already printed. Not required.**
@IdenticalCopies	**Use this field to specify how many label copies should print for each unique label. This feature is useful when printing labels with data from database or when you use counters, and you need label copies. Not required.**
@NumberOfSets	**Use this field to specify how many times the printing process should repeat. Each label set defines the occurrence of the printing process. For example: setting this value to 5 will cause the printing process to repeat five times. Not required.**
@Port	**Use this field to specify the port name for the printer. You can override the default port as specified in the printer driver. Not required.**
Other fields	**All other fields define the name of the variables from the label. The fields provide values for variables.**

The order or appearance of the columns in the CVS file is no important. But it is important that all rows in the same CVS file have the same structure.

Session print is used automatically. Session printing is disabled when you change the printer or label within the same CVS file.

All columns in the CVS file are used when setting the values to the label variables. If the variable with the name from CVS does not exist on the label, no error message is displayed.

Command File: XML File

The commands available in the XML command files are a subset from NiceCommands. You can use the following commands: Login, Quit, Label, LabelClose, Set, Port, Printer, SessionEnd, SessionStart, SessionPrint, SetDatabase and SetTable. Of course, the syntax of the commands differs a little bit when used in XML file.

The root element is **Nice_Commands** that must be present in the XML command file. The next element that must follow is **Label**, that specifies which label to use. Next you have two options for label printing:

1. Print labels normally using the element **Print_Job**.
2. Print labels in session using the element **Session_Print_Job**.

You can also change the printer to which the labels will print, you can set the variable value and you can change the database that is currently used on the label.

XML Command File Structure Explained

Below is the description of the command file structure. There are several elements that contain attributes. Some attributes are required, other are optional. Some attributes can occupy pre-defined values only, for other you can specify the custom values.

Nice_Commands: Is root element with two attributes

- login: Performs login procedure into the program. Not required.
- quit: Closes the print engine when the actions execute. Will remove the print engine from the memory. Is required.

Label: The element that opens the label in the print engine. If the label is already opened, it will be re-used. You can use this element several times within the command file.

- name: Attribute contains the label name. It is recommended to include label path and filename. Note: You can provide label name only, but be sure that print engine will try locate the label in the correct folder. Is required.
- close: The attribute instructs the print engine, if the label should close after printing, or it should remain open. Possible values: true, false. Is required.

Print_Job: The element that unions the commands for printing labels. You can use this element several times within the command file.

- printer: Use this attribute to override the printer defined on the label. Print the label to some other printer. The other printer must be accessible from this computer. Use the printer name for value of this attribute. Not required.
- quantity: Use this attribute to specify the number of labels to print. Possible values: numeric value, VARIABLE or UNLIMITED. Required.
- skip: Use this attribute to specify how many labels to skip at the beginning. This feature is useful if you print sheet of labels to laser printer, but the sheet is partial already printed. Not required.
- job_name: Use this attribute to specify the name of your job file. The specified name is visible in the print spooler. Not required.
- print_to_file: Use this attribute to specify the file name where you want to save the printer commands. Not required.

Session_Print_Job: The element that unions commands for printing labels. It considers session print rules. You can use this element several times within the command file. For available attributes lookup the attributes for the element **Print_Job**. All of them are valid, you only cannot use the quantity attribute. See the description of the element **Session** to find out how to specify label quantity in session printing.

Database: The element that overrides the database selection on the label. Use it whenever you do not want to use data from the database that is configured on the label, but some other database. You can use this element several times within the command file.

- name: The attribute contains the database name. Required.

Table: The element that overrides the table selection on the label. Use it whenever you do not want to use data from the table that is configured on the label, but some other table. You can use this element several times within the command file.

- name: The attribute contains the table name. Required.

Variable: The element that sets the value of variables on the label. You can use this element several times within the command file.

- name: The attribute contains the variable name. Required.

Command-line Parameters

Command-line Parameters

Command line parameters are used to pass advanced commands to the middleware integration server when program starts.

Parameter should be appended after program filename.

For NiceWatch in Standard Series:
C:\Program Files\EuroPlus\NiceLabel 5\Bin\NWatchS5.exe <parameter>

Note:
The command-line parameters are available for NiceWatch Server executable file NWATCHS5.EXE not with the NiceWatch manager executable file NWATCH5.EXE.

For NiceWatch Server **in the** Enterprise Series**:**
C:\Program Files\EuroPlus\NiceWatch Server 5\bin\nwatchsrvs5.exe **<parameter>**

Note:
The command-line parameters are available for NiceWatch Server executable file NWATCHSRVS5.EXE not with the NiceWatch manager executable file NWATCHSRV5.EXE.

Available parameters are:

/START	**Triggers are activated at startup.**
/STOP	**Triggers are deactivated at startup.**
[configuration file]	**Set the configuration file which will be used instead of the default one; i.e. C:\Triggers\NWatch.mis**

NiceCommands

NiceCommands

You can use NiceCommands to control label printing from the print engine from another program. NiceCommands are simple to use text commands. There are several NiceCommands available, but the usual are:

NiceCommand	Description
LABEL	**The name of the label to use. It is recommended to include label path and filename. Note: You can provide label name only, but be sure that print engine will try locate the label in the correct folder.**
PRINTER	**Use this command to override the printer defined on the label. Print the label to some other printer. The other printer must be accessible from this computer. Use the printer name for value of this attribute.**
PRINT	**Use this command to specify the number of labels to print. Possible values: numeric value, VARIABLE or UNLIMITED.**

SET	**Use this command to specify the value for the variables.**

COMMENT

;

When developing program code or scripts it is very wise to well document your commands. This will help you decode what the script really performs, when you will look at the code after some time.

Use semicolon (;) on the beginning of the line. Everything following it will be treated as script comment and will not be processed by application.

CREATEFILE

CREATEFILE <name_of_the_file>

This command will create a plain ASCII text file. The file will contain only one line of text.

The purpose of creating such file is to signal some external application that the label processing or printing has began or has ended.

The example of the CREATEFILE usage is printing labels with data from soem file. First the external application prepares variable data for the labels and store it into particular file. Then print engine is activated and printing starts. To inform the application when the printing process is finished, a file can be created on the disk. It could can be a signal to the application, that the printing application has an empty printing queue and new label printing can be started.

DELETEFILE

DELETEFILE <name_of_the_file>

This command deletes the specified file. You can use it in combination with CREATEFILE command.

EXPORTLABEL

EXPORTLABEL ExportFileName [, ExportVariant [, CreateLVXFile]]

The command is implemented to automate the "Export to printer" command. You can manually access the command using File -> Export -> Export to printer. The label is exported directly to the printer and stored in the memory for off-line printing. The user can recall the label with keyboard on the printer or sending a command file to the printer.

ExportFileName	**The parameter is mandatory and defines the filename of a generated print stream for exporting label to the printer.**
ExportVariant	**Some printers support multiple export variants. When manually**

	exporting, the user can select the export variant in the dialog. With the EXPORTLABEL command you must specify which export variant you want to use. **The first variant has the value 0. The second variant has the value 1, etc.** **If you do not specify the variant type, value 0 is used as default.**
CreateLVXFile	**The parameter has two possible values, TRUE or FALSE. If you set the value to TRUE, besides the generated print stream also the LVX file will be created. The LVX file contains information about the variables used in the label.** **For more information about exporting labels and LVX files refer to the white paper section on the website.**

LABEL

LABEL <name_of_the_file>

The command opens the working label. If the label is already opened, the program will use this one. It is recommended to write full path name along with the file name.

Note, if variable value contains space characters or commas, you will have to enclose the whole path in quotation marks (e.g. LABEL "C:\My Labels\sample3.lbl").

If you use LABEL command with the middleware integration server running in service mode, use UNC quotation instead of the mapped drives (e.g. LABEL "\\SERVER\SHARE\MY LABELS\LABEL.LBL" instead of "G:\MY LABELS\LABEL.LBL").

LABELCLOSE

LABELCLOSE

The command closes the currently active label. The label application will stay opened. FILECLOSE command does the same thing, but is depreciated. To speed up label printing do not use this command frequently. The label designer can have opened more label files simultaneously. If the label is already opened, it does not have to be loaded and thus the label processing can be performed quickly.

LOGIN

LOGIN <username>

Performs login procedure into the labeling program. This is necessary when login is required.

NOTE. This is a DDE command and should not be used in batch command .JOB files.

MESSAGEBOX

MESSAGEBOX message [, caption]

Displays the message in the message box. The second parameter is used to define the title of the message dialog box.

If the variable value contains space characters or commas, you have to enclose the text in quotation marks (e.g. MESSAGEBOX "Insert labels in printer", Warning).

OEMTOANSI

OEMTOANSI ON|OFF

This command works in conjunction with command SET. It puts the text that follows the command SET in proper codepage, so that variable is assigned the proper value.

Use it to put the values following SET command to the proper codepage, so correct values will be transferred to print engine at print time.

PORT

PORT <file_name>

This command overrides the printer's port name. This command is used to redirect print output to a file. In this case specify the name of the file in the parameter file_name. If the file path or filename contain space characters, enclose the whole string in double quotes.

PRINT

PRINT quantity [, skip [, identical label copies [, number of label sets]]]

Command PRINT starts printing. The first parameter is the quantity of the labels that should be printed.

<number>	**This many labels will be printed.**
VARIABLE	**Some variable contains the information how many labels should be printed. It can be label-defined prompted variable or a field from the database.**
UNLIMITED	**If you use a database to acquire values for variable fields, unlimited printing will print as many labels as there are record in the database. If you do not use a database, there is not much sense to use this option. In this case the maximum number of labels that thermal printer internally supports will be printed.**

The parameter skip in the command represents the number of the labels you want to omit before first printed label on the page. The parameter is used for label printing on sheets of paper. When the part of the page is already printed, you can re-use the same sheet by

shifting the start location of the first label. The rest of the unused labels on the page can be printed with the help of this parameter.

The parameter identical label copies specifies how many copies of the same label should be printed.

The parameter number of label sets specifies how many times the whole printing process should be repeated.

If you do not need to set some of the supplementary parameters, use their default values.

Skip	**0**
Identical label copies	**1**
Number of label sets	**1**

PRINTER

PRINTER <printer_name>

Normally, the PRINT command prints the label to the printer specified in the label file. Using this command you can override this defined printer and print the label to some other printer.

If the printer name contains space characters, you have to enclose it in quotation marks.

For printer_name always use the system printer name as is displayed in the status line in the label designer. System printer names are usually the same as the printer names in Printers folder from Control Panel, but not always, so pay attention. They differ only when you are using network-connected printers, when you should use "\\SERVER\SHARE" syntax and not a printer friendly name.

PRINTJOBNAME

PRINTJOBNAME <job_name>

Specifies the print job name that will be used in print manager when using PRINT command. After printing the name is returned in normal state. Use this option to easier distinguish between different printing jobs in the Windows spooler.

If variable value contains space characters or commas, you have to enclose the text in quotation marks (e.g. PRINTJOBNAME "Label for printing").

QUIT

This command stops the labeling program after printing. The application is closed.

RETURN

This command returns focus to the main labeling program after the printing completes.

SESSIONEND

The function closes data stream.

SESSIONPRINT

SESSIONPRINT quantity [, skip]

You send the data stream to printer using this function. You can use multiple SessionPrint commands one after another and join them in single data stream. The stream is not closed until the command SessionEnd occurs. The meaning of quantity and skip parameters is the same as with NiceCommand PRINT.

SESSIONSTART

All three commands(SessionStart, SessionPrint, SessionEnd) are used together. If ordinary command SessionPrint is used, every time a complete data stream for printer is sent. If you want to join multiple Print commands into one data stream, you can use the command SessionStart followed with any number of SessionPrint commands and in the end use the command SessionEnd. The stream is not closed until the command SessionEnd occurs.

These commands offer a way of optimal label printing. It is not necessary to generate a complete data stream for each print session, you can join more sessions in one stream.

SET

SET name=variable_value, [,step[, quantity_of_repetition]]

Name is the name of the variable defined on the label. If the variable isn't on the label, an error will occur. **Step** and **Quantity_of_repetition** are option parameter. These parameters tell the increment of the variable and the number of the labels before change.

If **Variable_value** contains space characters or commas, you have to enclose the text in the text qualifier marks. By default the text qualifier is a double quote character, but you can use any other character (refer to the command TEXTQUALIFIER).

If you want to assign multi-line value to a variable, use the syntax "\r\n" to encode newline character. "\r" is replaced with CR (Carriage Return) and "\n" is replaced with LF (Line Feed). Both, CR and LF, represent newline character in Windows operating system.

Note: Be careful when setting values to variables that provide data for pictures on the label, as backslash characters might be replaced with some other characters.

For example, if you assign a value "c:\My Pictures\raw.jpg" to the variable, the "\r" will be replaced with CR character and the final result is this:

```
c:\My Pictures
aw.jpg
```

SETDATABASE

SETDATABASE <database_name> = <value>

database_name	**The name of the currently used database as defined in the program.**
value	**The name of the new table that should be used as data source.**

This command allows you to use some other database with the label file and not the one, that was connected to the label file at design time.

This other database will only be used when printing labels, the label file will remain intact with connection to the original database.

SETPRINTPARAM

SETPRINTPARAM paramname=value

This command allows you to set advanced print parameters before printing.

Currently supported PARAMNAMES are:

PAPERBIN	**Use it to specify from which tray the paper should be used. If the printer is equipped with more than just one paper / label tray, you can control which is used for printing.** **The name of the tray should be acquired from the printer driver.**
PRINTSPEED	**Use this parameter so specify printing speed. The value for parameter varies from one printer to the other. Consult printer's manuals for numbers.**
PRINTDARKNESS	**Use this parameter so specify printing darkness / contrast. The value for parameter varies from one printer to the other. Consult printer's manuals for numbers.**

SETTABLE

SETTABLE <table_name> = <value>

table_name	**The name of the currently used table as defined in the program.**
value	**The name of the new table that should be used as data source.**

This command allows you to use some other table with the label file and not the one, that was connected to the label file at design time.

This other database table will only be used when printing labels, the label file will remain intact with connection to the original table.

The new database table should be of the same type as original table. For example, you cannot change the table from dBase to Paradox. The structure of new table has to be identical to the original one.

You can use table from the database that is already connected to the label or from some entirely different database.

TEXTQUALIFIER

TEXTQUALIFIER %

Text-qualifier is the character that embeds a data value that is assigned to a variable. If the data value includes space characters, it must be included in the text-qualifier. Otherwise only the data until the first encountered space character is assigned to the variable.

The default delimiter for the command SET is double-quote character. Because the double-quote character is used as shortcut for inch unit of measure, sometimes it is difficult to pass the data with inch marks in the JOB files.

The work-around is to use the double double-quote character instead of just one, but in this case already the incoming data stream needed to be changed. You can leave the incoming data stream as-is and change the delimiter you want to use.

For example:

TEXTQUALIFIER %

SET Var1 = % EPAK WRP BD 12"X10 7/32" %

The command TEXTQUALIFIER set the delimiter to percent sign (%). The command SET can then use new delimiter character (%) for specifying the value to the variable Var1.

Note:
The command TEXTQUALIFIER is persistent (during single program session). If your JOB file sets the TEXTQUALIFIER to some value, it will be used until set to another value.

Technical Support

Online Support

You can find the latest builds, updates, workarounds for problems and Frequently Asked Questions (FAQ) on the product web site at www.nicelabel.com. If you cannot solve the problem on your own, please contact your local vendor or representative offices listed in the topic **Contact Information**.

For more information please refer to:

- Support FAQ: http://kb.nicelabel.com
- NiceLabel FAQ: www.nicelabel.com/Learning-center/Sales-FAQ
- NiceLabel Tutorials: www.nicelabel.com/Learning-center/Tutorials
- NiceLabel Forums: forums.nicelabel.com

Contact Information

Head Office

Euro Plus d.o.o.

Poslovna cona A 2
SI-4208 Šenčur
Slovenia
Tel: +386 4 280 50 00
Fax: +386 4 233 11 48
support@nicelabel.com
www.nicelabel.com

North American Office

Niceware International, LLC.
200 South Executive Drive, Suite 200
Brookfield, Wisconsin 53005
USA
Tel: (888) 894-NICE (6423)
Fax: (262) 784-2495
support@nicewareintl.com
www.nicewareintl.com

German Office

NiceLabel Germany GMBH
Liebknechtstr. 29
63179 Obertshausen
Germany
Tel: +49 (0)6104 405 400
support@nicelabel.de
www.nicelabel.de

China Office

GARDEN BUSINESS CENTER
Julu Rd. 889, Building 23
Office 8311
Shanghai 200040, PR China
Tel: +21-62490371
Fax: +21-62490372
support@nicelabel.cn
www.nicelabel.cn