BarTender®
LABEL & RFID SOFTWARE

White Paper

Commander

Utility for Cross-Platform and Enterprise Integration

SEAGULL SCIENTIFIC

Contents

An Overview of How Commander Works

Commander is a software utility, provided with the Enterprise editions of BarTender, that enables you to perform automatic label printing using BarTender in situations where it is not convenient or possible to perform automation using ActiveX or command lines. Commander can be run as an application or as a Windows service.

Triggers

When a software application needs to generate labels, it simply performs a triggering action, such as placing a file in a location of your choosing on the network, transmitting an e-mail, or sending information through a TCP/IP socket connection. Commander detects the arrival or occurrence of these "triggers" and then "wakes up" BarTender so it can merge your data into the label design and automatically print your labels.

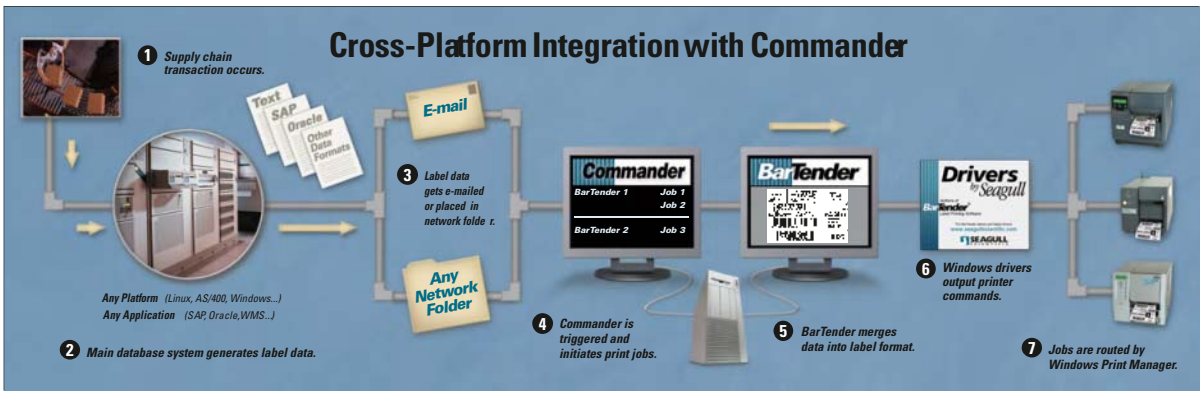
Some Different Trigger Types

A trigger file or message can be an empty "wake up call" without data. Alternatively, the trigger can contain label data and script commands that Commander, BarTender, and/or another application that Commander launches.

As an example, assume that an in-house order fulfillment program saves order information into a database *and* creates a trigger file named NewOrder.dat in a directory being watched by Commander. After Commander detects the creation of that file, it would "wakes up" BarTender and the label job could be processed in a number of ways, including:

- **If the Trigger File is Empty**: If it is decided that the trigger file or message will not contain data, then the BarTender label format might be configured to query the database for information about orders entered after a specified time and use the query results in the label print job.
- **If the Trigger File Contains Label Data**: Alternatively, the BarTender print job might be set up to simply read the label data right out of the trigger file.
- **Other Types of Trigger Content**: In addition to label data, trigger files and messages can also contain Commander Script and BarTender XML Script.

Regardless of the trigger type, once the print job is done being processed, Commander would usually be configured to delete the NewOrder.dat trigger file so that it could resume monitoring of the directory and wait for the next label job.

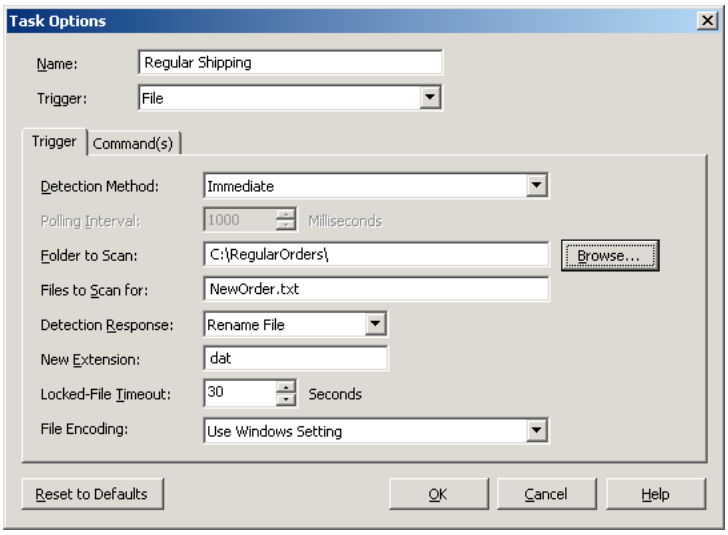


Tasks

A Commander *task* is a set of one or more commands associated with a trigger.

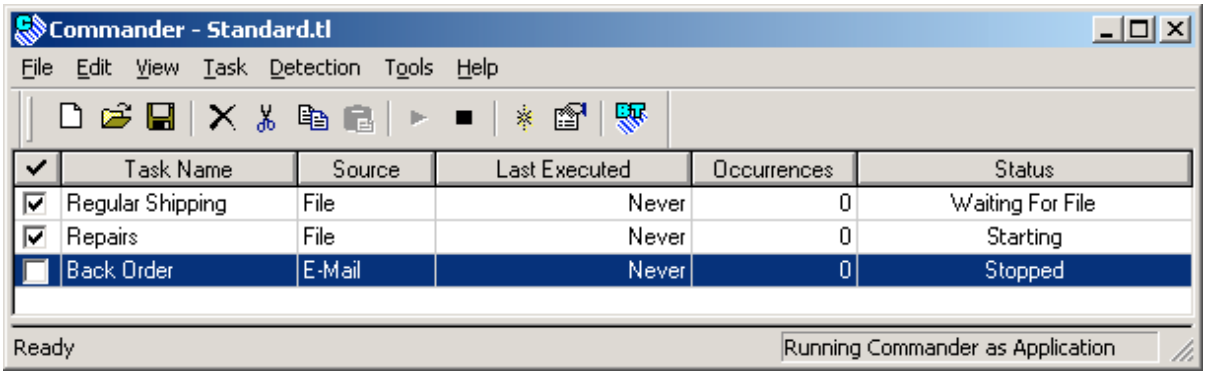
There are two major parts to defining a task in Commander:

- First you select what type of trigger will cause Commander to execute the command(s) for that task and you specify certain options.
- Then you define the actual commands that will be executed in response to the trigger. These can be any combination of BarTender commands, operating system command lines, and Commander script.



Task Lists

Commander can monitor more than one trigger and execute a separate task in response to each one. The task or set of tasks that Commander is ready to execute at any given time is called a *task list*. You may define and save more than one task list, but only one can be active in Commander at any one time



How Commander Detects Triggers

Commander's methods of detecting triggers are customizable. For example, Commander can detect file triggers using either of two methods:

- **Interrupt-driven:** Commander detects almost immediately that the trigger file has been created.
- **Polling-driven:** Commander checks the target directory at periodic intervals for the presence of the trigger file. You can set the interval with precision measured in milliseconds.

There are also multiple ways that TCP/IP Socket communications can trigger Commander, such as:

- Arrival of a specific sequence of characters.
- Occurrence of predefined periods of inactivity.
- Disconnection of the socket's source.

Using Commander to Run Any Application

Commander's primary purpose is to allow software programs to print labels with BarTender without actually having to directly control BarTender. However, Commander can launch *any* system command or installed Windows application that can be executed using the **Run** option of the Windows **Start** menu. Therefore, Commander can respond to a received trigger by doing much more than just print BarTender labels.

For example, assume that a company's customer service department processes customers by having them come to web site to fill out forms requesting assistance and that the entered data is then transmitted to headquarters in the form of an e-mail. A copy of Commander running at headquarters could be configured to trigger when those emails arrive, write the e-mail contents to a text file, and then call an application that reads the file and stores its contents in

a database. This is an example of how Commander's trigger-based integration technology can simplify your middleware challenges.

Advanced Commander Capabilities

Overview

Here are some more complex triggering scenarios that Commander can handle:

- It does not matter whether the application creating the trigger is running on Windows or not.
- The wildcard character * and **?** can be used to specify the trigger file name (or the Subject line when an e-mail trigger is used). This lets you easily take advantage of serialize file names (such as name001, name002, etc.) so that triggers can occur faster than Commander can process them.
- Special Commander variables can be used so that Commander can fill in certain values at task execution time and pass the data to BarTender.
- The contents of a trigger can contain the command line parameters that BarTender is to use when launched.
- The trigger contents can contain the data that should appear on a label.
- Commander can be configured to preload copies of BarTender and selected label formats in advance. This way, the execution of tasks does not have to be slowed down waiting by these loading actions but can start printing labels right away.
- Commander can perform more than one command in response to a given trigger.
- Commander can instruct BarTender to close after it has printed the label.
- Commander can be set to log all of its activities.

Commander Scripts

A Commander script is a set of written instructions that Commander can read and execute. Applications that create triggers can add include script commands in the triggers. This allows the application to give Commander instructions that vary from trigger-to-trigger, instead of having Commander perform exactly the same actions in response to every trigger of a given type.

The primary purpose of a Commander script is to specify a BarTender command line and to specify the data that is to populate the fields on the label.

Commander Variables

Commander lets you specify variables on the BarTender command line, operating system command line, and in Commander scripts. Commander then substitutes appropriate values for these variables when the commands in a task are executed. Among other uses, these variables can specify parameters such as label format name, printer name, text file name, IDoc file name, and ODBC user name and/or password.

Variables available for replacement at command execution time include:

Commander Variable	Substitution at Command Execution Time
<Trigger Contents>	Replaced by the contents of the trigger.
<Trigger File Name>	Replaced by the complete path and filename of the trigger file.
<Trigger Base File Name>	Replaced by the root trigger filename. (Does not include either the path or the extension.)
<Trigger File Path>	Replaced by the trigger file's path. (Does not include the filename or extension.)
<Time>	Replaced by the time the task was triggered. (Uses the format [h]h:mm:ss xM.)
<Date>	Replaced by the date the task was triggered. (Uses the format [m]m/[d]d/yyyy.)
<E-mail Subject>	Replaced with the entire Subject line of the e-mail message. (For use with e-mail triggers only.)

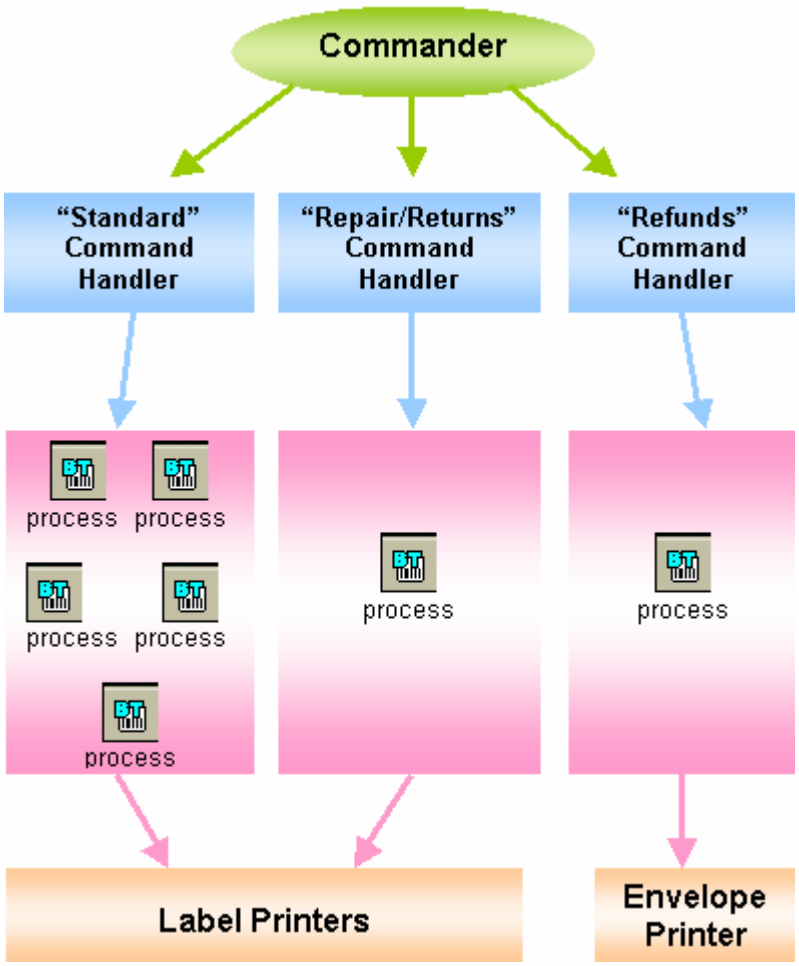
BarTender Command Handlers

You can increase the speed at which labels are printed by having multiple instances (or "processes") of BarTender running, each working simultaneously on the same or different tasks. Furthermore, different instances of BarTender can launch with different sets of command parameters so you can assign each command of a task to the most appropriately-configured instance of BarTender. You can also save system resources by having seldom-triggered tasks immediately close BarTender after it is done being used to execute a task.

Each command in a task can specify a "Command Handler" to execute it. A Command Handler is a named group of settings that specify command line parameters to use when starting BarTender, as well as the number of BarTender processes to start. Command Handlers are primarily responsible for preloading label formats and passing print job instructions to BarTender processes. With the Enterprise Print Server Edition (described below), if a particular Command handler is heavily used, you can define it to use more than one BarTender process. By default, all tasks share a single Command Handler. This minimizes the use of system memory, but does not perform as well in some cases if custom Command Handlers are used.

For example, a user could create these three command handlers:

- Assume that because the command handler called "Standard" is used to create labels for a high-volume shipping application, it preloads the same label format into five different BarTender processes and leaves them running. Each time a new order is placed, a trigger is created by the shipping application and Commander uses the first available BarTender process to print the label. (The ability to assign multiple BarTender processes to a Command Handler is specific to the Enterprise Print Server edition of Commander.)



- Assume that a second command handler, called "Repair/Returns," is used less frequently. It is therefore sufficient for it to preload and use only one BarTender process.
- Finally, assume that a third command handler, called "Refunds," is assigned to a task that is triggered only a few times a week to print an envelope. In order to conserve system resources, the Command Handler does not preload a BarTender process or label

format. Instead, it loads BarTender and the label format only as needed, prints the required envelope, and then has Bartender exit.

Commander Features added with Enterprise Print Server Edition

A number of new, advanced capabilities were added to Commander to support requirements of the Enterprise Print Server edition of BarTender. These capabilities are not available with the Enterprise and RFID Enterprise editions.

The Commander features added for the Enterprise Print Server edition include:

TCP/IP Socket Triggers

In addition to triggering on Files and Emails, Commander Print Server is capable of triggering on TCP/IP sockets. As with trigger files, TCP/IP socket communication can simply be an "empty" trigger, or the communication can also contain label data and/or Commander Script code. TCP/IP sockets are a good option when a file share or file creation is not possible or convenient. Direct TCP/IP socket communication is also faster than the creation of intermediary data files and allows superior programmatic interaction between the originating application and Commander.

Ability to Submit BarTender XML Script

The Print Server Edition of BarTender supports an XML-based scripting language that gives applications even more control of BarTender than Commander Script offers. Whereas Commander executes Commander Script directly and performs the requested actions itself, Commander passes on received BarTender XML Script for execution by BarTender.

XML Transforms

If XML is received by Commander as part of a trigger, Commander can use XSL transforms to convert it into any other data format. One of the best uses of this feature is to allow applications to output XML in their own familiar format and then depend on Commander to convert it into the XML script readable by BarTender.

Multiple BarTender Processes

With the Enterprise Print Server edition of Commander, it is possible to assign multiple BarTender processes to a single Command Handler (described above, in the "Command Handler" section). This allows for more rapid response to large numbers of closely occurring Commander triggers, such as might be encountered in high-demand centralized printing environments.

For More information

For more details on Commander, please consult:

- The "Command Examples" white paper.
- The Help system inside of Commander.

For more details on BarTender XML Script, please consult:

- The Help system inside of BarTender.

Available Seagull White Papers

General White Papers

- The Advantage of Drivers by Seagull
- Choosing the Right BarTender Edition
- What's New in the Latest BarTender

Integration White Papers

- Integration Overview
- Getting Started with ActiveX Automation Using C#
- Getting Started with ActiveX Automation Using VB.NET
- Getting Started with ActiveX Automation Using VB6
- Commander
- Commander Examples
- Exporting Printer Code Templates
- Using BarTender with Terminal Services and Citrix MetaFrame
- XML Integration with Oracle's WMS and MSCA

SAP Integration White Papers

- SAP Integration Methods
- Reading SAP IDocs

Miscellaneous White Papers

- BarTender Enterprise Licensing
- Printing International Characters Using BarTender
- BarTender Software Activation
- Using BarTender's Application Identifier Wizard
- Optimizing Label Printing Performance
- Status Monitor Overview

For downloadable versions, visit:

www.seagullscientific.com/aspx/whitepapers.aspx

SEAGULL SCIENTIFIC

www.seagullscientific.com