opentext™

OpenText™ Intelligent Capture

Utilities Modules Guide

This guide contains information about the Intelligent Capture utilities client modules.

ECPCORE220300-CMU-EN-1

OpenText™ Intelligent Capture
Utilities Modules Guide
ECPCORE220300-CMU-EN-1
Rev.: 2022-June-13

This documentation has been created for OpenText™ Intelligent Capture CE 22.3.
It is also valid for subsequent software releases unless OpenText has made newer documentation available with the product, on an OpenText website, or by any other means.

Open Text Corporation

275 Frank Tompa Drive, Waterloo, Ontario, Canada, N2L 0A1

Tel: +1-519-888-7111
Toll Free Canada/USA: 1-800-499-6544 International: +800-4996-5440
Fax: +1-519-888-0677
Support: https://support.opentext.com
For more information, visit https://www.opentext.com

Table of Contents

Chapter 1

Utilities Modules

Starting in release 22.3, this new guide contains a compilation of the Intelligent Capture utilities client modules. In previous releases, these were available as separate guides.

For information about common features and functionalities in the Intelligent Capture client modules, see *OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

Table 1-1: Utilities Modules

Module	Original module ID	Link
.NET Code	ecpcore-cnt	See ".NET Code Module"
Copy	ecpcore-ccb	See "Copy Module"
Multi	ecpcore-cpu	See "Multi Module"
Timer	ecpcore-cti	See "Timer Module"

Chapter 2

.NET Code Module

This section includes the Intelligent Capture utilities module: **ecpcore-cnt**.

2.1 Introduction

The .NET Code Module enables a developer to add custom behaviors to a CaptureFlow process. This means that a process can perform tasks although other batch processing modules do not enable these tasks. For example, you can set up the module to read information from an external file or database. You can also set the module to restructure the batch tree according to custom business logic.

The module provides a Microsoft .NET programming interface that can be used to read and write batch data. A developer accesses this interface by creating a .NET assembly (DLL file). The .NET programming environment also provides access to built-in .NET Framework interfaces. This means that the process can use system objects such as files and security objects directly.

The .NET Code Module is incorporated into a process using *OpenText Intelligent Capture - Designer Guide (ECPCORE-CPD)*, in the same way as any other module step. The module can run as a service and automatically process any tasks it receives.

Notes

- You can also use the .NET Code Module in a process created with Process Developer. Trigger it by setting the <*Ready*> value to any non-zero value data.
- The .NET Code Module should be run as a service for normal production work. Application mode can be used for administrative and debugging purposes.

2.2 Setting Up .NET Code

Most configuration information for Intelligent Capture client modules is defined during *module setup*. You can run a module in setup mode from Intelligent Capture Designer, from Intelligent Capture Administrator, or from a command prompt with appropriate arguments. For more information, refer*OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

When launched in setup mode, a client module displays the **Setup** window in which you can specify the module's configuration settings.

2.2.1 Creating a Custom Code Assembly

This section explains how to write your own custom code for use with the .NET Code Module.

2.2.1.1 Setting up a Custom Code Project

You create a custom code assembly as a .NET project.

To create a custom code project

1. Open Microsoft Visual Studio.
2. Create a class library based on the .NET Framework, specifying Visual Basic, or Visual C# as the programming language. The *Intelligent Capture Release Notes* (available in My Support (https://support.opentext.com/)) contains information on supported versions of .NET Framework.

   See the Visual Studio documentation for detailed instructions on creating a project.
3. Add a project reference to the assembly file `Emc.InputAccel.CaptureClient.dll`. This file can be found in the client `binnt` subfolder.

   See the Visual Studio documentation for detailed instructions on adding a project reference.
4. Create a class that inherits from the base class Emc.InputAccel.CaptureClient.CustomCodeModule.

   For example, the following Visual Basic code creates a custom code module class named MyCustomModule:

   ```
   Imports Emc.InputAccel.CaptureClient

   Public Class MyCustomModule
       Inherits CustomCodeModule
       ' CLASS PROPERTIES AND METHODS GO HERE
   End Class
   ```
5. Within your custom code module class, create a public default constructor. Here is how it would look in Visual Basic:

   ```
   Public Sub New()
   End Sub
   ```
6. Within your custom code module class, create methods that override the following base methods:

   - *`StartModule`*

     This method is called as soon as the first task is started after the module opens. It is passed a reference to an interface that enables you to set and retrieve information about the .NET Code Module.

   - *`ExecuteTask`*

This method contains the code that performs your custom actions. It runs each time the module receives a new task. For the first task received, this method is called immediately after `StartModule`.

- The `ExecuteTask` function must end by calling either the `CompleteTask()` method or the `FailTask()` method of its `IClientTask` parameter. The name of this parameter in the base class is *`task`*. For a simple implementation with no error checking, add `task.CompleteTask()` as the last line of code within `ExecuteTask`.

Note: If the function does not call either `CompleteTask()` or `FailTask()`, the module throws an exception and returns code IA_ERR_RETRYSOME (-6113) to the workflow server.

The following Visual Basic code shows a valid `ExecuteTask` method override:

```
Public Overrides Sub ExecuteTask(ByVal task As IClientTask, ByVal batchContext As
IBatchContext)
    ' CODE THAT PERFORMS CUSTOM ACTIONS GOES HERE
    task.CompleteTask()
End Sub
```

Example 2-1: “Hello, World” Custom Module

Here is a simple custom code module assembly that writes a message to a custom log file. This code is the project code in Visual Basic:

```
Imports Emc.InputAccel.CaptureClient
Imports System.IO

Public Class MyCustomModule
    Inherits CustomCodeModule

    Public Sub New()
    End Sub

    Public Overrides Sub ExecuteTask(ByVal task As IClientTask, ByVal batchContext As
IBatchContext)
        Using outputFile As StreamWriter = New StreamWriter("C:\logfile.txt")
            outputFile.WriteLine("Hello, world!")
            outputFile.Close()
        End Using
        task.CompleteTask()
    End Sub

    Public Overrides Sub StartModule(ByVal startInfo As ICodeModuleStartInfo)
    End Sub

End Class
```

Here is the same sample project written in Visual C#:

```
using Emc.InputAccel.CaptureClient;
using System.IO;

public class MyCustomModule : CustomCodeModule
{

    public MyCustomModule()
    {
    }

    public override void ExecuteTask(IClientTask task, IBatchContext batchContext)
```

```
    {
        using(StreamWriter outputFile = new StreamWriter(@"C:\logfile.txt"))
        {
            outputFile.WriteLine("Hello, world!");
            outputFile.Close();
        }
        task.CompleteTask();
    }

    public override void StartModule(ICodeModuleStartInfo startInfo)
    {
    }

}
```

When you compile this custom assembly and assign it to a .NET Code Module step for a particular process, the module writes the message to the output file for every task it receives.

Note: A module running as a service cannot display a graphical user interface or collect runtime input from the module operator. Because the .NET Code Module most typically run as a service, avoid adding custom code that attempts to perform these operations. Doing so could cause the module to lock up and prevent the further processing of tasks.

Related Topics

"Setting up a Custom .NET Code Module Step" on page 16

"Accessing Batch Data" on page 12

"Controlling Batch Flow" on page 14

2.2.1.2 Accessing Batch Data

A custom code assembly can access the data in the batch it is processing. You can retrieve information about the batch, and you can also modify the batch. For example, you could reorganize the pages in a document.

You can retrieve a batch node from the `ExecuteTask` method in the following ways:

- Through the BatchNode property of the *`task`* parameter. This property enables access to the portion of the batch tree that is passed to the step in the CaptureFlow process. The node retrieved by this property is at the level defined for the step. If the step is defined at level 7, the property retrieves the full batch.

  Typically a module is expected to work with batch data at the level defined in the CaptureFlow process.

- Through the *`batchContext`* parameter. This parameter provides access to the entire batch.

You can access the entire batch regardless of the level at which the .NET Code Module step is defined. This access is useful to retrieve information about the batch as a whole while processing lower-level nodes.

In each case, you can use the properties and methods of the object to set and retrieve batch data.

Example 2-2: Setting a Node Value

The following code sets the custom value <*UserID*> to the Windows user name of the logged on user:

```
batchContext.GetRoot("CustomValues").NodeData.ValueSet.WriteString("UserID",
System.Environment.UserName)
```

Example 2-3: Getting a Page Count

The following Visual Basic code retrieves the number of pages in the entire batch (as <*batchPageCount*>). It also retrieves the number of pages in the task being processed (as <*taskPageCount*>). The `GetDescendantNodes` method does not apply to nodes at level 0, so the code checks the node level before attempting to use this method. A task at level 0 always contains only one page.

This example assumes that your custom code step is called “Code”.

```
Dim batchPageCount As Integer = 0
Dim taskPageCount As Integer = 0
Dim taskRoot As IBatchNode

' GET NUMBER OF PAGES IN ENTIRE BATCH
batchPageCount = batchContext.GetRoot("Code").GetDescendantNodes(0).Count

' GET NUMBER OF PAGES IN CURRENT TASK
taskRoot = task.BatchNode
' CAN ONLY CHECK DESCENDENTS FOR CONTANER LEVELS
If (taskRoot.RootLevel > 0) Then
    taskPageCount = taskRoot.GetDescendantNodes(0).Count
Else
    ' TASK IS RECEIVED AT PAGE LEVEL
    taskPageCount = 1
End If
```

The following is the equivalent code written in Visual C#:

```
int batchPageCount = 0;
int taskPageCount = 0;
IBatchNode taskRoot;

// GET NUMBER OF PAGES IN ENTIRE BATCH
batchPageCount = batchContext.GetRoot("Code").GetDescendantNodes(0).Count;

// GET NUMBER OF PAGES IN CURRENT TASK
taskRoot = task.BatchNode;
// CAN ONLY CHECK DESCENDENTS FOR CONTANER LEVELS
if (taskRoot.RootLevel > 0)
{
    taskPageCount = taskRoot.GetDescendantNodes(0).Count;
}
else
{
```

```
    // TASK IS RECEIVED AT PAGE LEVEL
    taskPageCount = 1;
}
```

Related Topics

"Controlling Batch Flow" on page 14

"Creating a Custom Code Assembly" on page 10

2.2.1.3 Controlling Batch Flow

In a custom code assembly, you use the *task* parameter of the `ExecuteTask` method to indicate status to the CaptureFlow process. This parameter has the following methods that help you control task flow:

- The `CompleteTask` method indicates that your custom code successfully completed its task. The system is free to move the appropriate batch data to the next step of the process.
- The `FailTask` method indicates that your custom code completed, but it encountered a problem while performing the task. When you call this method, you pass it information about the nature of the problem, including a reason code. What happens after that depends on the error handling logic that is configured in the process. What happens if the module encounters an error outside of your custom code also depends on this error handling logic.

Call one of these methods when your code has finished its custom processing. If you do not, the task is canceled and an error is logged.

Example 2-4: Failure to Read from a File

You want to fail the task of reading data from an external file during execution of your custom assembly. You want a warning if the data cannot be read. The following Visual Basic code demonstrates how to fail the task:

```
Dim value As String = ""
Try
    Using inputFile As StreamReader = New StreamReader("C:\values.txt")
        value = inputFile.ReadLine()
        inputFile.Close()
    End Using
Catch ex As Exception
    task.FailTask(FailTaskReasonCode.ResourceUnavailable, ex)
End Try
```

The following is the equivalent code written in Visual C#:

```
string value = "";
try
{
    using(StreamReader inputFile = new StreamReader(@"C:\values.txt"))
    {
        value = inputFile.ReadLine();
        inputFile.Close();
```

```
    }
}
catch (System.Exception ex)
{
    task.FailTask(FailTaskReasonCode.ResourceUnavailable, ex);
}
```

Related Topics

"Accessing Batch Data" on page 12

"Creating a Custom Code Assembly" on page 10

2.2.1.4 Creating Custom Setup Options

You can create custom setup controls for a custom code assembly. The custom setup controls are available when the .NET Code Module is run in setup mode. With custom setup controls, you create a more flexible custom assembly. Its behavior can change without requiring a change to the assembly code itself.

To create custom configuration options, create a `SetupCodeModule` method inside your CustomCodeModule class. This method is called when the **Setup** button is clicked in the module setup for a process or batch that uses this custom assembly. If the custom assembly does not have a `SetupCodeModule` method, then the custom setup option is not available.

Example 2-5: Prompting for a Custom Setup Value

The following is an implementation of the `SetupCodeModule` method written in Visual Basic. This example displays a prompt for an input string that is then assigned to a custom value. If the user cancels the prompt or supplies an empty string, the method returns false to indicate that no changes are necessary.

```
Public Overrides Function SetupCodeModule(ByVal parentWindow As Control, ByVal
stepConfiguration As IValueAccessor) As Boolean
    Dim saveResult As Boolean = True
    Dim enteredValue = ""
    enteredValue = InputBox("Enter custom value:")
    If (enteredValue = "") Then
        saveResult = False
    Else
        stepConfiguration.WriteString("customValue", enteredValue)
    End If
    Return (saveResult)
End Function
```

Related Topics

"Running the Module for Setup" on page 16

2.2.2 Setting up a Custom .NET Code Module Step

Once a custom code assembly is created, it must be assigned to a .NET Code Module step in a process to run.

2.2.2.1 Running the Module for Setup

A .NET Code Module step must be configured with information about which .NET class performs the custom actions for this step. This configuration can be done through administrative tools, or by running the module directly using setup command line arguments.

Setup is normally configured at the process level, and any batch created using that process inherits the process design settings. It is also possible to change the design settings for a specific batch after that batch has been created. Batch-specific changes to design settings do not affect any other batches based on the same process.

To run the module for setup from a command line

1. Write your custom assembly.
2. Run the module for setup.
3. Under **Choose the code assembly to use**, select the custom assembly file.
4. Under **Choose the class that implements the desired functionality**, select the name of the .NET class that performs the custom actions.

   Only classes that inherit from Emc.InputAccel.CaptureClient.CustomCodeModule are listed here.
5. If the custom class is configured with additional setup options, click **Setup** to access those options. This portion of setup is entirely customized, so what you see is an interface created by the custom code itself. If you need help using this custom interface, contact the developer who created the custom assembly.
6. When you return to the main setup window, click **OK** to save your changes to the setup options.

2.2.2.2 Deploying a Custom Assembly

Before your custom code can be used to process batches, it must be deployed to the server using Intelligent Capture Designer.

To deploy a custom code assembly

1. Create a `\bin` folder under the `<CaptureSystem>` directory if it does not exist.
2. Copy the DLL and data files to `<CaptureSystem>\bin`.
3. Copy any additional data files to the same folder.

4. Enter the names of the DLL and all data files as a comma-separated list in the **DeploymentFiles** global option for the **Capture System** in Intelligent Capture Designer.

5. Deploy the files to the server from Intelligent Capture Designer.

6. Stop and restart the .NET Code Module service.

   You can use Windows Control Panel to stop and restart the service. See the Windows documentation for detailed information about managing services.

Notes

- If the custom code assembly changes and you update it, restart the module before the changes take effect.
- Suppose you are not running the module as a service but you are running it as an application for debugging or administrative purposes. If the .NET Code Module is not running as a service, only exit the module window and reopen it.

Related Topics

"Running the Module for Setup" on page 16

"Debugging and Error Handling" on page 17

2.2.3 Debugging and Error Handling

This section describes methods you can use to identify and fix problems that can occur when the .NET Code Module runs.

2.2.3.1 Custom Code Errors

You can attempt to resolve errors that occur when custom code runs in the following ways:

- Custom code can be written to detect and respond directly to problems that can be encountered when the code runs.
- You can create a debug trace log to write information about errors to a file.

Note: The .NET Code Module should be run as a service for normal production work. Application mode can be used for administrative and debugging purposes.

Related Topics

".NET Code Module Errors" on page 20

2.2.3.1.1 Writing Code to Handle Errors

Custom code can be written to detect and respond to errors that can occur when the code runs.

- Ordinary safe coding practices include validating external input before using it. It is possible to catch most errors this way and take appropriate action in the custom code.

  From the perspective of the custom code, "external input" is anything that comes from outside the custom code assembly—including batch data. As an example, suppose you wanted to assign specific value data to the second page of every batch. Before making such an assignment, check that the batch actually contains at least two page nodes.

- Unanticipated errors are known as *exceptions* in .NET, and there are built-in mechanisms for handling them. The `try…catch` statement in Visual Basic or Visual C# provides a mechanism for the code itself to detect and respond to exceptions.

  See the Visual Studio documentation for detailed information about exception handling.

- After performing a custom task, the custom code can verify that the processing occurred as expected. If it did not, the code can call `IClientTask.FailTask` to indicate to the system that the batch step was not successfully completed. This call completes the custom code processing and results in a module error.

- For runtime debugging, you can use the `ICodeModuleStartInfo.Trace` methods to indicate that you want to write debug information to a log file. Run the module with a command line argument that specifies where to write the information. "Creating a Debug Trace Log" on page 19 explains how to debug information to a log file.

Related Topics

"Creating a Debug Trace Log" on page 19

".NET Code Module Errors" on page 20

"Controlling Batch Flow" on page 14

2.2.3.1.2 Creating a Debug Trace Log

You can write runtime debug information from the .NET Code Module to a log file on the client system. Run the module with the *-trace* command line argument, as in the following example:

```
Codeclient.exe -tracelevel:debug
```

Writing a Custom Message to the Trace Log

A custom code assembly can call the `ICodeModuleStartInfo.Trace` methods to write custom debug messages to the trace log. An optional exception parameter writes exception information to the actual message.

Example 2-6: Writing a message to the trace log

The following Visual Basic example writes a trace message indicating a problem retrieving custom data from an XML file. The file name is passed as a runtime parameter in the message.

```
Dim xmlFileName = "file://c:\data.xml"
Dim xmlData As System.Xml.XmlReader
Try    xmlData = System.Xml.XmlReader.Create(xmlFileName)
Catch ex As System.Exception
    startInfo.Trace(ex, "The custom XML data could not be read from {0}.", xmlFileName)
End Try
```

The following is the equivalent code written in Visual C#:

```
string xmlFileName = @"file://c:\data.xml";
System.Xml.XmlReader xmlData;
try
{
    xmlData = System.Xml.XmlReader.Create(xmlFileName);
}
catch (System.Exception ex)
{
    startInfo.Trace(ex, "The custom XML data could not be read from {0}.", xmlFileName);
}
```

Related Topics

"Writing Code to Handle Errors" on page 18

".NET Code Module Errors" on page 20

2.2.3.2 .NET Code Module Errors

Error handling for the .NET Code Module errors that occur outside of the custom code assembly is defined in the CaptureFlow process. For information about configuring error handling in a process, see *OpenText Intelligent Capture - Designer Guide (ECPCORE-CPD)*.

Notes

- If the configured custom code assembly or the class is invalid or is not found, the task fails and returns error code IA_ERR_BADSETUP (-4404) to the server.
- Some error conditions cease to exist when subsequent attempts are made to process a task. For example, a temporarily locked file can cause an error initially. This error does no longer exist by the time the module attempts to process the task again.

2.3 Running .NET Code in Production

After the module has been set up, tested, and placed in a production environment, operators and administrators will run the module in production.

For information related to common production tasks for unattended modules, see *OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

Note: The *Running Modules in Production Mode* (*OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*) contains production topics that are common to many unattended modules and may not apply to this module.

2.4 Reference—.NET Code

The topics within this section contain reference information useful while using the application in setup or production.

2.4.1 Keyboard Shortcuts and Access Keys

The .NET Code Module provides access keys that facilitate navigation through various module windows, options, and panels.

Access keys are available to increase accessibility. Press the **ALT** key to reveal access keys indicated as underlined alphanumeric characters. Access keys generally affect only the currently active window, pane, or panel. Standard Windows shortcuts can be used to navigate between windows, panes, or panels.

2.4.2 IA Values

The topics in this section describe IA values that can be used with this application.

2.4.2.1 Input IA Values

.NET Code Module has no input values.

Related Topics

"Output IA Values" on page 21

2.4.2.2 Output IA Values

The .NET Code Module outputs data to statistical values during processing, as described in the following table.

Statistical IA Values

<*ErrorText*>
The error message resulting from task failure, or "Canceled" if the task was canceled. If the task was successful, this value is blank.

- **Attributes**: String, Output
- **Level**: T

<*StartDateTimeUtc*>
The date and time at which processing started for the task, in Coordinated Universal Time (*UTC*). The system clock of the client is used to determine the time. If task processing never started, this value will be set to a year prior to 2000.

- **Attributes**: Date, Output
- **Level**: T

<*TaskExecutionStatus*>
A numerical code representing the result of the task:

- -1 = The task failed. Check the <*ErrorText*> value for additional information.
- 0 = The task was not processed.
- 1 = The task was completed successfully.
- **Attributes**: Long, Output
- **Level**: T

<*TotaltimeMilliSec*>
The total task processing time, in milliseconds.

This is the total amount of time that the module was actively processing the task.

- **Attributes**: Long, Output
- **Level**: T

2.4.3 Programming Reference

.NET Code module scripting is done using the `Emc.InputAccel.CaptureClient` namespace. For the complete programming reference, see *OpenText Intelligent Capture - Scripting Guide (ECPCORE-PSC)*.

Interfaces are provided as .NET libraries. Custom code is written using standard .NET programming languages such as Visual Basic and Visual C#.

Chapter 3

Copy Module

This section includes the Intelligent Capture utilities module: **ecpcore-ccb**.

3.1 Overview

The Copy module is used to copy Intelligent Capture batches for either remote processing or remote storage of batches as follows:

Any Intelligent Capture Server

The Copy module connects to a single source Intelligent Capture Server (which can be in a ScaleServer group) and sends batch files to another single destination Intelligent Capture Server (optionally selected from a list of servers or from within a ScaleServer group) for additional processing; the Copy module can also copy a batch from a remote server.

Notes

- Although a Copy module can only connect to a single source Intelligent Capture Server, multiple Copy modules can connect to the same source Intelligent Capture Server.
- Multiple Copy modules can run on the same machine.

Any local or network directory available from your machine

The Copy module creates backup files of batches. The copied batch file is compressed in ZIP format to save disk space and is unzipped when the batch is restored to an Intelligent Capture Server.

An FTP site

The Copy module stores batch files on a remote network to save disk space on your machine. Copied batch files are compressed in ZIP format to save disk space and then unzipped when the batch is restored to an Intelligent Capture Server.

Note: Because the Copy module is designed to copy batches only, it does not copy any associated runtime files, which are external to the batches. Examples of these runtime files are profiles, document types, projects, and DLLs. Use Intelligent Capture Designer to deploy these runtime files to the Intelligent Capture Server to which you are copying the associated batches. For more information, see *OpenText Intelligent Capture - Designer Guide (ECPCORE-CPD)*.

3.1.1 Processing Modes for Tasks

The Copy module processes tasks in one of two processing modes:

Copy Batch mode
Copies a single batch or files from the Intelligent Capture Server, remote server, directory, or *FTP* site.

Run All Batches mode
Configured during setup, **Run All Batches** mode is used to schedule the batch copying function from the Intelligent Capture Server to another server, directory, or FTP site.

The Copy module always runs at level 7 (batch level), because it is designed to process tasks that consist of all the nodes in a batch.

3.1.2 Copy Module Features

Features of the Copy module include the following:

Provides batch copy support for multiple steps in a process
Insert multiple Copy steps into a single Intelligent Capture process. For example, you can write a process that scans a batch on an Intelligent Capture Server in Texas, copies and sends the batch file to an Intelligent Capture Server in Barbados for *OCR* and indexing, and then copies and sends it back to the Intelligent Capture Server in Texas for export.

Copies selected stage files or the *IAB* file separately
Useful when copying large batch files. It also helps identify problems that occurred in a particular part of your batch.

Saves a processing log
Save a copy of the current log file to review it and diagnose problems with batch processing.

Uses a file naming schema to generate batch file names
When copying batch files to a directory, Copy can insert IA Values into file names. Inserting these values enables the batch files to be saved and organized into subdirectories according to processing level.

Note: Users running the Copy module must be assigned the **Server.Copy.Batch.to.Server** permission through Intelligent Capture Administrator. This permission enables users to copy batches to the Intelligent Capture Server.

3.2 Setting Up Copy

Most configuration information for Intelligent Capture client modules is defined during *module setup*. You can run a module in setup mode from Intelligent Capture Designer, from Intelligent Capture Administrator, or from a command prompt with appropriate arguments. Find the details in *OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

When launched in setup mode, a client module displays the **Setup** window in which you can specify the module's configuration settings.

3.2.1 Specifying Batch Copy Settings

Use the **Copy Settings** window to define the source and destination as well as file compression parameters for files to be copied.

3.2.1.1 Specifying Source Settings

Specify the settings for the source from where the batch is copied.

To specify the source settings for copying a batch:

1. Run the Copy module in setup or production.
2. Click **Copy Settings** in setup mode or **Copy Batch** in production mode. The **Copy Settings** window displays.
3. Click the **Source** tab.
4. Select one of the following options:
   - **Directory**: Use to specify settings to retrieve batch from a local or network directory
     - **File name**: type the file name and click the **Browse** button. When retrieving files from a directory, you can use wildcards (`*` or `?`) in the **File name** field to specify more than one file at a time. For example, *.ZIP would retrieve all ZIP files in the indicated directory (e.g., `C:\IA\batchfiles\*.zip`).
     - **Stop task and report error if**: When selected, enables the check boxes for reporting errors. When the check box is selected and the associated error occurs, an error message displays. If cleared, Copy will log an error, skip processing the current file, and continue processing the next file. The available options are:
       - **No files were found**
       - **A zip file was unable to decompress**
       - **A process was not uploaded**

- **Unable to activate a process after uploading**
- **Unable to delete a zip file**

- **Server**: Use to specify the source server when copying files from a server, or retrieving files from a server
  - **Server**: The name of the server where the batch or files currently reside.
  - **User**: Your user name for the specified server.
  - **Password**: Your password for the specified server.
  - **Batch name**: The name of the batch or process.
  - **Copy only the following stage files**: The **Browse** button displays the **Select Files to Copy** window, where you can select the files to be copied. For more information, see Copying stage files only.
  - **Copy domain user/group permissions**: Copies domain batch permissions from the source Intelligent Capture Server to a destination Intelligent Capture Server. For more information, see Copying batch permissions.
  - **Copy local user/group permissions**: Copies batch permissions that have been granted in local users and groups.
- **FTP**: Use to specify settings to retrieve a batch from an FTP site.
  - **Location**: Specifies the location of the *FTP* server. You can type in the location or file name, or click the **...** button to display the **Choose FTP Location** window to specify the FTP location. If you type the FTP server and file name (e.g. `ftp://ftp.server.com/mybatch.zip`), you can use wildcards (* and ?) in your file names to specify related files. For example, `ftp://ftp.server.com/*.zip` would process all zip files in the specified location.
  - **User**: Your user name for the FTP server.
  - **Password**: Your password for the FTP server.

3.2.1.2 Specifying Destination Settings

You can specify the settings for the destination to where the batch is copied.

To specify the destination settings for copying a batch:

1. Run the Copy module in setup or production.
2. Click **Copy Settings** in setup mode or **Copy Batch** in production mode. The **Copy Settings** window displays.
3. Click the **Destination** tab.
4. Select one of the following options:

- **Directory**: Use to specify settings for the destination for your batch when copying to a directory.
  - **File name**: Type the file name or select the file using the browse button.

    Note: Destination zip file names for cannot contain special characters (- _ ' ' ~ # ! @ $ % [ ] { } ( ) +) .
  - **Insert Value**: Displays the **Choose Value** window for inserting IA Values in file names.
  - **If file exists**: If the file of the same name exists in the directory, you can specify one of these options for how to handle the situation.
    - **Overwrite**: Overwrites a file that has the same name. Copy processes the batch and the Copy log displays an error message.
    - **Skip**: Stops processing the current task when a file with the same name exists. Copy moves on to the next available task. The Copy log displays a warning. To finish processing the skipped task, retrigger the batch through Intelligent Capture Administrator.
    - **Abort**: If run as an application, this option will stop the Copy module from processing the current batch when a file with the same name as the copied batch exists. If run as a service, the Copy module will continue processing tasks when a file with the same name as the copied batch exists.
    - **Prompt**: If run as an application, the module displays a prompt when a file with the same name exists. At the prompt, click **Yes** to overwrite the file or **No** to stop processing the current batch. If run as a service, the Copy module will continue processing tasks when a file with the same name as the copied batch exists.
- **Server**: Use to specify the destination server when "Copying Batch Files From One Intelligent Capture Server to Another Intelligent Capture Server" on page 35.
  - **Server**: The name of the destination server.
    - For a single server, specify the server name.
    - To enable the selection of an available server from among several servers, specify a list of servers, wherein each server name is separated by a semicolon. For example:

      ```
      servername1;servername2;servernameN
      ```

      Note: The **User** and **Password** credentials provided must be valid for all the specified Intelligent Capture Servers.
    - To enable the selection of an available server from a ScaleServer group, specify the name of any server in the ScaleServer group followed by a semicolon. For example:

```
LondonIAserver;
```

Note: If multiple destination servers (or a ScaleServer group) are specified, the module randomly selects a server in the list (or the ScaleServer group) until it successfully logs on. The batch is then copied to that server. The Copy module uses the credentials specified on the **Destination** tab in setup mode for all servers.

- **User**: Your user name for the specified server.
- **Password**: Your password for the specified server.
- **Batch name**: The name of the batch or process.
- **If file exists**: If the file of the same name exists on the destination server, you can specify one of these options for how to handle the situation.
- **Overwrite**: Overwrites a file that has the same name. Copy processes the batch and the Copy log displays an error message.
- **Skip**: Stops processing the current task when a file with the same name exists. Copy moves on to the next available task. The log displays a warning that the task was skipped because a file with the same name exists. To finish processing the skipped task, the batch can be retriggered through Intelligent Capture Administrator.
- **Abort**: When the copy module is run as an application, stops processing the current batch when a file with the same name as the copied batch exists. When the module is run as a service, it continues processing tasks so that no user intervention is required. It finishes the retry task with `IA_ERR_EXISTS` and then sets the batch priority to zero.
- **Prompt**: In application mode, displays a prompt when a file with the same name exists. At the prompt, click **Yes** to overwrite the file or **No** to stop processing the current batch. In service mode, the behavior is the same as described for the **Abort** option

- **FTP**: Use to specify settings for the FTP server to receive the copied files.
  - **Location**: Specifies the location of the *FTP* server. Type in the location or file name, or browse to display the **Choose FTP Location** window to specify the FTP server location.
  - **User**: Your user name for the FTP server.
  - **Password**: Your password for the FTP server.

5. Click **OK** to save the specified **Destination** settings.

3.2.1.3 Specifying Additional Copy Options

You can specify additional options that involve specifying the log file definition for the copy operations and batch compression parameters.

To specify additional options for copying a batch:

1. Run the Copy module in setup or production.
2. Click **Copy Settings** (in setup mode) or in production mode. The **Copy Settings** window displays.
3. Click the **Options** tab.
4. Select the **Enable logging** check box to save a copy of the log file. The log file receives statistical information and error messages for processed batches.

   Note: Selecting this option saves the log file for all batches processed through Copy, not just the specified batch.

5. Specify the **File path**where the log file must be saved. This field is available when you select the **Enable logging** check box. Enter the directory path and file name of the log file to save. You can also click the **Browse** button to the right of the field to open the**Save As** window. Navigate to the directory, type a name for the log file, and click **Save**.
6. Select the **Log level** to determine the level of detail in the log file.

   - Level 1 is the least detailed logging level. It includes all errors and some general information.
   - Level 5 is the most detailed level and contains all information from lower levels plus debugging information.
   - Level 1 or 2 is sufficient for the most normal use of the module and provides good economy of log file space.
   - Levels 4 and 5 are primarily intended for debugging problems and usually are only used when trying to resolve problems. Higher levels fills the log file more quickly.

   Note: When the log file reaches its maximum size of 5 MB, entries wrap around and begin overwriting the oldest entries in the file.

7. Select the **Convert to JBIG** check box to enable Copy to compress binary image files in the copied batch using *JBIG* compression. This option does not produce JBIG files, but only uses JBIG compression. This option is useful for saving disk space.

   Note: This check box is available only when copying a batch to a directory or an *FTP* site.

8. Select the **Delete source when finished** check box to enable Copy to delete the original file from the source when copying is complete. To select this option, you must have permission to delete the batch or file.

   Note: This check box is unavailable when copying a file from an FTP site.

9. Click **OK** to save the specified **Destination** settings.

3.2.1.4 Specifying the FTP Location for FTP Source and FTP Destination Settings

The **Choose FTP Location** window is used to specify the FTP location either when specifying source settings or specifying destination settings.

To specify the FTP location for source and destination settings

1. Run the Copy module in setup or production.
2. Click **Copy Settings** (in setup mode) or in production mode. The **Copy Settings** window displays.
3. Click the **Source** tab to specify source settings or the **Destination** tab to specify destination settings.
4. Select the **FTP** option. Click the **...** next to the **Location** field to display the **Choose FTP Location** window.
5. Specify the **FTP Server** name in the format <*ftp.servername.com*>.
6. Enter the **Username** and **Password** to access the selected server.
7. Select **Passive FTP Mode** to enable passive mode file transfers between the client and the remote *FTP* site.
8. Click **Connect** to display a list of the directories available on the specified FTP server.
9. In the **Location** field, type a directory name or select a directory from the list displayed.
10. Click **Disconnect** to disconnect from the currently active FTP server and then specify a new FTP server.
11. Click **OK** to save the FTP location settings.

3.2.2 Specifying a Copy Schedule for a Department

If the Copy module step in your process has a department associated with it, the you must specify a schedule . If a schedule is not specified, the department does not receive any tasks for the Copy module.

To specify a copy schedule for a department

1. Access the **Schedule** window. To do so, do one of the following:
   - Run the Copy module in setup mode. Click **Scheduling** to display the **Schedules** window. Click **Add** to display the **Schedule** window.
   - Run the Copy module in production mode. From the Run menu, select **Scheduling Options** to display the **Schedules** window. Click **Add** to display the **Schedule** window.
2. Enter the department name for the departments specified in the Copy step in the *IPP* in the **Dept. Name** field.
3. Specify the **Time** to **Start** and **Stop** the copying of queued batches.
4. Specify the copying **Frequency**:
   - **Every day**: Starts and stops at the specified times every day.
   - **Selected days**: Starts and stops only on the selected days. Enable the check box next to the day you want to perform the copy operation.
   - **Selected dates of each month**: Starts and stops only on the specified date of each month. Enter the numeric date or dates on which the copy occurs.
5. Click **OK** to save the Schedule settings.

3.2.3 Managing Currently Scheduled Copy Operations

The **Schedules** window displays a list of the currently scheduled automatic copy activities for a department. Use this window to specify additional schedules or edit the current list of schedules.

To manage currently scheduled copy operations:

1. Access the **Schedules** window. To do so, do one of the following:
   - Run the Copy module in setup mode. Click **Scheduling** to display the **Schedules** window.
   - Run the Copy module in production mode. From the **Run** menu, select **Scheduling Options** to display the **Schedules** window.

   The **Schedules** window displays information about copy schedules currently available on the server. Select a schedule from this list before clicking the **Edit** or **Delete** buttons.

2. Do one of the following:

   - Click **Add** to specify a copy schedule for a department.
   - Select an existing copy schedule and click **Delete** to remove the schedule.
   - Select an existing copy schedule and click **Edit** to display the **Schedule** window where you can change parameters for the copy schedule.

3. Click **OK** to save changes made to the list of copy schedules.

3.2.4 Working with User Permissions

The Copy module enables you to copy batch permissions from the source Intelligent Capture Server to a destination Intelligent Capture Server. The Copy module uses Microsoft Windows permissions to allow or deny copying of batch permissions. If users are defined on the destination server or are in the domain to which the server belongs, then the u.User batch permissions are copied to the destination server. Before assigning user permissions, ensure that:

- The user specified on the **Copy Settings** window **Destination** tab has change permissions on the `IAS\Batches` folder, and all of its subfolders and files, on the destination Intelligent Capture Server.
- If the user specified on the **Destination** tab is a local user on the destination server, the user has administrator rights or change permissions on the `IAS\Batches` folder of the destination server.

3.2.4.1 Assigning User Permissions

If users are defined on the destination server or are in the domain to which the server belongs, then the user batch permissions are copied to the destination server. You must have administrator rights on the destination server to carry out the following procedure. This procedure is a Windows procedure and is performed outside of Intelligent Capture.

To assign the required permissions to a local, non-administrator:

1. On the destination server, use Windows Explorer to locate the `IAS\batches` folder, or the folder where your batches are located.
2. Select the batches folder, then choose**Properties** from the **File** menu to display the **Batches Properties** window.
3. Select the **Security** tab, then click the **Advanced** button to display the **Advanced Security Settings For Batches** window.
4. Select the local user in the **Permissions Entries** list, then click the **Edit** button to display the **Permission Entry For Batches** window.
5. Select the **Allow** check box for **Change Permissions**.
6. Set the **Apply Onto** field to **This Folder, Subfolders, and Files**.

7. Click the **OK**button to apply the permission and close the window.
8. Ensure that **Change Permissions** appears in the **Permission** column of the **Advanced Security Settings For Batches** window.
9. Click **OK** on each of the remaining permissions windows.

Caution

Granting **Change Permissions** permission to a user grants that user considerable power including the ability to change permissions of all batches. (It does not affect security in other folders.) Consider the security implications carefully before proceeding.

Note: A non-*NTFS* installation does not support batch security.

3.2.5 Handling Error Codes During Copy Requeuing

By design, the Copy module receives a task twice to process it properly:

- The first time it receives a task, Copy returns the error code -6114 (IA_ERR_RETRY). This error causes requeuing of the task on the appropriate Intelligent Capture Server.
- The second time it receives the task, Copy processes the task normally, and either returns a success status (IA_SUCCESS) or an error code (other than -6114). If Copy returns an error the second time it processes the task, the priority of that batch is set to zero. Doing this prevents the batch from being returned to Copy for continued reprocessing when a genuine error occurs.
- If you include an `Error` event handler in your *IPP* for Copy, retrigger the first task by setting <*RetriesLeft*> to a value greater than zero. You can ignore error code -6114 in your Windows Event Log.

Note: If Copy copies the batch to a zip file, Intelligent Capture Export is not triggered until Copy restores the zip file on an Intelligent Capture Server.

3.3 Running Copy in Production

After the module has been set up, tested, and placed in a production environment, operators and administrators will run the module in production.

For information related to common production tasks for unattended modules, see *OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

Note: The *Running Modules in Production Mode* section contains production topics that are common to many unattended modules and may not apply to this module.

3.3.1 Copying Batches

The Copy module is used to copy batches for either remote processing or remote storage of batches. Each batch contains all necessary processing instructions, the page files to be processed, and the data that results from processing.

Notes

- Destination zip file names cannot contain special characters (- _ ' ' ~ # ! @ $ % [ ] { } ( ) +) .
- If Copy is installed on both source and destination servers then departments must be used to specify batch routing. If a department is not specified, then the Copy module on the destination server receives tasks that are intended for Copy on the source server.

3.3.1.1 Copying in Copy Batch Mode

Copy Batch mode copies only a single batch or file at a time from your Intelligent Capture Server, and is the only way to copy files from a remote server, directory or *FTP* site. In **Copy Batch** mode, you can only copy individual batches from your Intelligent Capture Server because you can only specify one file at a time. You configure **Copy Batch** settings in production mode, and after the batch is processed the specified settings are not saved. When the batch is copied from a server, it is not removed from the source server and remains intact and unaltered unless you specifically set up deletion of source files.

To process tasks in Copy Batch mode:

1. Run the Copy module. The **Copy** window displays.
2. Click the **Copy Batch** button or select **Copy Batch** from the **File** menu. The **Copy Settings** window displays.
3. When you finish configuring the **Copy Settings** window, click **OK**. Copy automatically performs the copy tasks you configured in the **Copy Settings** window.
4. To cancel processing, click **Stop**. Copy does not finish copying the batch and displays an error message in the log file.
5. When Copy finishes copying the batch, the **Run All Batches** and **Copy Batch** processing buttons redisplay and a finish message displays in the log file.
6. Select one of the processing mode buttons to continue exporting batches.
7. Select **Exit** from the **File** menu to quit Copy. Copy logs off from the Intelligent Capture Server and exits.

3.3.1.2 Copying in Run All Batches Mode

Use the **Run All Batches** mode to process all tasks that are queued on a single Intelligent Capture Server.

Note: The Copy module can only log in to and process a task from a single server; that is, the Copy module does not support connecting to multiple source servers.

To copy in Run All Batches mode:

1. Run Copy for production.
2. Click **Run All Batches**. The machine receives tasks from batches as they become available from the Intelligent Capture Server. **Run All Batches** is a **Wait for Task** mode, which means that only batches queued on the Intelligent Capture Server for the specific module can be processed.
3. To stop **Wait for Task** mode, select the **Stop** command.
4. To close the module, select **Exit** from the **File** menu.

3.3.1.3 Copying Batch Files From One Intelligent Capture Server to Another Intelligent Capture Server

Copy enables you to copy and send a batch file to an Intelligent Capture Server. This function is useful for sending files for processing at a remote Intelligent Capture Server.

You can copy a batch from one server to another server in either Run All Batches mode or Copy Batch mode. See those particular procedures for more information on running the modules in those modes. This topic explains the particulars of specifying source and destination servers.

Note: The Intelligent Capture Server page count is incremented when using the `IATreeCommitPages` function. However, the Copy module does not use this function. If documents are scanned on one Intelligent Capture Server and then copied to another Intelligent Capture Server using the Copy module, the page count on the second server is not decremented.

To copy a batch from one Intelligent Capture Server to another server:

1. In the **Copy Settings** window, select the **Source** tab and select **Server**. Your Intelligent Capture Server information displays in the **Server**, **User**, and **Password** fields.
2. If you are configuring a batch or process, the batch or process name appears next to **Batch Name** by default. If you are configuring settings in Copy Batch mode, select a batch to be copied.
   - Click **Browse**. The **Open Batch** window displays.

- Select the batch to be copied and click **OK**. The batch displays next to **Batch name**.

3. Select the **Copy domain user/group permissions** check box to copy batch permissions that have been granted in a Windows domain.
4. Select the **Copy local user/group permissions** check box to copy batch permissions that have been granted in local users and groups.
5. If necessary, assign the required permissions to a local user on the destination server. For instructions, see Copying batch permissions.
6. Select the **Destination** tab and select **Server**.

   Type information for the destination of the copied batch in the **Server**, **User**, and **Password** boxes.
7. Select one of the **If file exists** options:

   - **Overwrite**: Overwrites a file that has the same name.
   - **Skip**: Stops processing the current task when a file with the same name exists.
   - **Abort**: Stops processing the current batch when a file with the same name as the copied batch exists.
   - **Prompt**: Displays a prompt when a file with the same name exists.
8. Select the **Options** tab and specify any settings.
9. After you have configured all setup options, click **OK** to save your settings to the Intelligent Capture Server and close the **Copy Settings** window.

3.3.1.4 Copying a Batch File to a Local or Network Directory

With **Copy** you can automatically or manually back up batches on a local or network directory. The copied batch file is compressed into a `ZIP` format to save disk space and is unzipped when the batch is restored to an Intelligent Capture Server. You can also unzip these backup files without restoring them to the Intelligent Capture Server.

Note: When copying a batch file to a directory, Copy enables you to insert data from an IA Value into the file name. This feature enables you to save your batch file in separate subdirectories based on the node with which each page is associated. Using file naming schemas makes it easy to identify and retrieve data from large batches.

You can copy a batch to a local or network directory in either Run All Batches mode or Copy Batch mode. See those particular procedures for more information on running the modules in those modes. This topic explains the particulars of specifying directories that will receive the copies.

To specify the directory information for copying a batch:

1. From the **Copy Settings** window, select the **Source** tab, then select the **Server** option. Fill in the **Server**, **User**, and **Password** fields appropriately.

   - If you are setting up a batch or process step, the batch or process name appears next to **Batch Name** and cannot be changed.
   - If you are configuring a Copy Batch operation in production mode, select a batch to copy. Click the **Browse** button next to the **Batch name** field to display the **Open Batch** window. Select the batch to copy and click **OK**. The selected batch name now appears in the **Batch name** field.

2. Set other **Source** tab settings as needed.

3. Select the **Destination** tab, then select **Directory**.

4. In the **File name** field, type the directory path and file name for the copied file. Alternatively, click the **Browse** button. The **Save As** window displays. Navigate to the directory, type a name for your new file, and click **Save**. Focus is returned to the **Copy Settings** window.

   Note: Destination zip file names for cannot contain special characters as follows: `(- _ ' ' ~ # ! @ $ % [ ] { } ( ) +) .`

5. Select one of the **If file exists** options:

   - **Overwrite**: Overwrites a file that has the same name.
   - **Skip**: Stops processing the current task when a file with the same name exists.
   - **Abort**: Stops processing the current batch when a file with the same name as the copied batch exists.
   - **Prompt**: Displays a prompt when a file with the same name exists.

6. Select the **Options** tab and specify any settings. From here, you can select the **Convert to JBIG** check box to compress binary image files in the copied batch using *JBIG* compression.

7. After you have configured all setup options, click **OK** to save your settings to the Intelligent Capture Server.

8. To retrieve the batch file and restore it to an Intelligent Capture Server, see Retrieving batch files from a directory.

3.3.1.5 Copying a Batch File to an FTP Site

Copy enables you to copy and save a batch file to an *FTP* site. This feature enables you to save batch files to a remote network without using disk space on your machine. The copied batch file is compressed in ZIP format to save disk space and is unzipped when the batch is restored to an Intelligent Capture Server.

You can copy a batch to an FTP site in either Run All Batches mode or Copy Batch mode. See those particular procedures for more information on running the modules in those modes. This topic explains the particulars of specifying FTP sites that will receive the copies.

To specify the directory information for copying a batch:

1. From the **Copy Settings** window, select the **Source** tab to and select the **Server** option. Your Intelligent Capture Server, user name, and password display by default in the **Server**, **User**, and **Password** fields.

   - If you are configuring a batch or process, the batch or process name appears in the **Batch Name** field.
   - If you are configuring settings in Copy Batch mode, select a batch to copy. Click the **Browse** button next to the **Batch name** field to display the **Open Batch** window. Select the batch to copy and click **OK**. The selected batch name now appears in the **Batch name** field.

2. Set other **Source** tab settings as necessary.
3. Select the **Destination** tab, then select **FTP**.
4. In the **Location** field, type the directory path and file name for the copied file. Alternatively, click the**Browse** button. The **Choose FTP Location** window displays. Select the location from the list box, or type in the **FTP Server information**, the **Username**, and **Password**. Click **OK** and focus is returned to the **Copy Settings** window.
5. Select the **Options** tab and specify any settings. From here, you can select the **Convert to JBIG** check box if you want Copy to compress binary image files in the copied batch using *JBIG* compression.
6. After you have configured all setup options, click **OK**.
7. To retrieve the batch file and restore it to an Intelligent Capture Server, see Retrieving batch files from an FTP site.

Note: When Copy copies a batch to an FTP site, the batch is compressed in ZIP format. You also can unzip these backup files without restoring them to the Intelligent Capture Server.

3.3.1.6 Copying Batch Permissions

The Copy module enables you to copy batch permissions from the source Intelligent Capture Server to a destinationIntelligent Capture Server. The Copy module uses Microsoft Windows permissions to allow or deny copying of batch permissions. If users are defined on the destination server or are in the domain to which the server belongs, user batch permissions are copied to the destination server.

Note: A non-*NTFS* installation does not support batch security.

You can copy batch permissions in either Run All Batches mode or Copy Batch mode. See those particular procedures for more information on running the modules in those modes. This topic explains the particulars of copying permissions.

To copy batch permissions:

1. Assign the required Windows permissions to a local user on the destination server. This step is required before you proceed with the following steps. For more information, see Assigning user permissions on the destination server.
2. In the **Copy Settings** window, select the **Source** tab and click **Server**. Your Intelligent Capture Server information displays in the **Server**, **User**, and **Password** fields.
3. Specify settings as described in "Copying Batch Files From One Intelligent Capture Server to Another Intelligent Capture Server" on page 35.
4. To specify batch permission, select **Copy domain user/group permission** to copy batch permissions that have been granted in a Windows domain. In addition, you can select **Copy local user/group permissions** to copy batch permissions that have been granted in local users and groups.
5. Select the **Destination tab** and click **Server**.
6. Type information for the destination of the copied batch in the **Server**, **User**, and **Password** boxes.
7. Select one of the **If file exists** options:
   - **Overwrite**: Overwrites a file that has the same name.
   - **Skip**: Stops processing the current task when a file with the same name exists.
   - **Abort**: Stops processing the current batch when a file with the same name as the copied batch exists.
   - **Prompt**: Displays a prompt when a file with the same name exists.
8. Select the **Options** tab and specify settings as needed.
9. After you have configured all setup options, click **OK** to save your settings to the Intelligent Capture Server and close the **Copy Settings** window.

Caution

The **Copy Local User/Group Permissions** option poses a potential security risk, because a local account on one machine is a different entity from a local account on another machine. For example, it is possible that a user may have different batch permissions on the source server than on the destination server, and using this option will override those permissions. Therefore, it is not recommended that you use this option if security is an issue.

Granting the **Change Permissions** permission to a user grants that user considerable power: the ability to change permissions of all batches. (It does not affect security in other folders.) Consider the security implications carefully before proceeding.

3.3.1.7 Copying Stage Files Only

Copy enables you to copy only selected stage files or only the *IAB* file, rather than copying an entire batch file (an IAB file and its corresponding stage files). This feature is useful when you are copying large batch files. A stage file is an output file produced by a module. Each module that processes pages receives stage files from the previous module. The receiving module creates a stage file. Stage files are stored on the Intelligent Capture Server for the life of the batch.

You can copy stage files in either Run All Batches mode or Copy Batch mode. See those particular procedures for more information on running the modules in those modes. This topic explains the particulars of copying stage files.

Caution

Use this option carefully. This feature is primarily used for diagnosing problems with a batch file or to save disk space by saving only one set of stage files from a batch. To continue processing a batch on the destination Intelligent Capture Server, the entire batch file is required.

To copy stage files:

1. In the **Copy Settings** window, select the **Source** tab and select **Server**. Your Intelligent Capture Server information appears by default in the **Server**, **User**, and **Password** fields.
2. Specify settings as described in "Copying Batch Files From One Intelligent Capture Server to Another Intelligent Capture Server" on page 35.
3. Select the **Copy only the following stage files** check box if you want to copy only specific stage files within a batch or to copy only the IAB file from a batch.
   - To copy only the IAB file (no stage files), leave the field empty.
   - To copy only specific stage files (with IAB file), click the **Browse** button. The **Select Files to Copy** window displays.

4. Click **Add**. The **Choose Value**window displays, enabling you to select the IA Value or file you want to copy. Click **OK**. Repeat for each IA Value or file to copy.
5. After you have selected the stage files to copy, click **OK** to close the **Select Files to Copy** window. Focus is returned to the **Copy Settings** window.
6. Select the **Destination** tab and click **Server**, **Directory** or **FTP**, depending on where you want to copy the stage files.
7. Select one of the **If file exists** options:
   - **Overwrite**: Overwrites a file that has the same name.
   - **Skip**: Stops processing the current task when a file with the same name exists.
   - **Abort**: Stops processing the current batch when a file with the same name as the copied batch exists.
   - **Prompt**: Displays a prompt when a file with the same name exists.
8. Select the **Options** tab and specify any settings.
9. After you have configured all setup options, click **OK** to save the stage files to the location you specified and close the **Copy Settings** window.
10. To restore and reprocess these files later, however, you must have the entire batch file.

Related Topics

"Copying a Batch File to a Local or Network Directory" on page 36

"Copying Batch Files From One Intelligent Capture Server to Another Intelligent Capture Server" on page 35

"Copying a Batch File to an FTP Site" on page 38

3.3.1.8 Copying Data to Intelligent Capture Server 6.x

Copy enables you to copy data from a previous version of Intelligent Capture Server to Intelligent Capture 6.x. Because the Intelligent Capture 6.x uses different data formats, the Intelligent Capture Server converts the data to be compatible with Intelligent Capture 6.x during the Copy process.

Note: If a server upgrade is performed, data residing on an Intelligent Capture Server is automatically converted and no special procedures are required. This procedure applies only when moving or copying *VBA*-based processes or batches from a previous version of Intelligent Capture Server to an Intelligent Capture Server 6.x. It cannot be used to copy *PCF*-based batches to Intelligent Capture 6.x, or to copy data from Intelligent Capture 6.x to an earlier version of Intelligent Capture Server.

To copy data from an existing Intelligent Capture 5.x system to Intelligent Capture 6.x:

1. Run the Copy module for production on the Intelligent Capture (or earlier) system. The **Copy** window displays.
2. Click the **Copy Batch** button, or select **Copy Batch** from the **File** menu. The **Copy Settings** window displays.
3. Select the **Source** tab and enable the **Server** option.
4. With your cursor in the **Server**: field, type the connection information for the Intelligent Capture Server 5.x.
5. Click the **Browse** button to display the **Open Batch** window. Select the batch to copy and click **OK**. The batch is displayed next to **Batch name**.
6. Select the **Destination** tab and enable the **Directory** option.
7. Click the **Browse** button and select a file name.
8. Click **OK** to copy the batch to the file system.
9. Run the Copy module for production on the Intelligent Capture Server 6.x .
10. Select the **Source** tab and enable the **Directory** option.
11. Click the **Browse** button to select a file name.
12. Select the **Destination** tab and enable the **Server** option.
13. With your cursor in the **Server** field, type the connection information for the Intelligent Capture Server 6.x .
14. Click **OK** to copy the batch to the server.
15. Repeat these steps for each batch you want to copy to the Intelligent Capture Server 6.x.

3.3.2 Restoring Batches

Copy enables you to retrieve batches from a directory or an *FTP* site and save the file to an Intelligent Capture Server. This feature is useful for restoring batch files to an Intelligent Capture Server from backup batch files on a network directory or FTP site. The backup files are saved in ZIP format and are unzipped when they are restored to an Intelligent Capture Server.

Note: You can also unzip these backup files without restoring them to the Intelligent Capture Server.

3.3.2.1 Retrieving Batch Files from a Directory

When batch files are backed up to a directory, the copied batch file is compressed in ZIP format to save disk space. When retrieving the file, it is unzipped when restored to an Intelligent Capture Server. You can retrieve one or more files from a directory during this restore operation. This task is only available in **Copy Batch** mode.

To retrieve a batch file from a local or network directory and restore it to an Intelligent Capture Server:

1. Run the Copy module. The **Copy** window displays.
2. Click **Copy Batch** or select **Copy Batch** from the **File** menu. The **Copy Settings** window displays.
3. Select the **Source** tab, and click **Directory**.
4. In the **File name** field, type the directory path and file name for the file you want to copy. You can use wildcards (* and ?) in your file names to specify related files (for example, *.zip would process all zip files in the directory). Alternatively, click the **Browse** button. The **Select File to Open** window displays. Navigate to the directory and select the file to copy , and then click **Open**.
5. Select any of the**Stop Task and report error if** check boxes if you want the Copy module to display a window describing the error for the selected options and abort the current task. If the check box for a particular error is cleared, Copy will log an error, skip processing the current file, and continue processing the next file.
6. Select the **Destination** tab. The **Server** option is selected by default.
7. Type information for the destination Intelligent Capture Server in the **Server**, **User**, and **Password** fields.
8. Select one of the **If file exists** options.
9. Configure any **Options** tab parameters, as necessary.
10. After you have configured all setup options, click **OK** to close the **Copy Settings** window and launch the retrieval process.

Related Topics

"Retrieving Batch Files from an FTP Site" on page 44

3.3.2.2 Retrieving Batch Files from an FTP Site

When batch files are backed up to an *FTP* site, the copied batch file is compressed in ZIP format to save disk space. When retrieving the file, it is unzipped when restored to an Intelligent Capture Server. You can retrieve one or more files from an FTP site during this restore operation. This task is only available in Copy Batch mode.

To retrieve a batch file from an FTP site and restore it to an Intelligent Capture Server:

1. Run the Copy module. The **Copy** window displays.
2. Click **Copy Batch** or select **Copy Batch** from the **File** menu. The **Copy Settings** window displays.
3. Select the **Source** tab, and click **FTP**.
4. In the **Location** field, type the FTP server and file name (e.g. `ftp://ftp.server.com/mybatch.zip`). You can use wildcards (* and ?) in your file names to specify related files (for example, `ftp://ftp.server.com/*.zip` would process all zip files in the specified location). Alternatively, you can click the browse button to display the **Choose FTP Location**window.
5. Type your **User name** and **Password**.
6. Select the **Destination** tab. The **Server** option is selected by default.
7. Type information for the destination Intelligent Capture Server in the **Server**, **User**, and **Password** fields.
8. Select one of the **If file exists** options.
9. Configure any **Options** tab parameters, as necessary.
10. After you have configured all setup options, click **OK** to close the **Copy Settings** window and launch the retrieval process.

Related Topics

"Copying a Batch File to a Local or Network Directory" on page 36

"Copying a Batch File to an FTP Site" on page 38

"Retrieving Batch Files from a Directory" on page 43

3.3.3 Saving a Copy of the Log File

When the Copy module is running in Copy Batch mode perform the following steps to save a copy of the current log file. When you copy a batch, the log file is populated with the log information.

To save a copy of the log file:

1. Run Copy for production. The module must be running in Copy Batch mode.
2. Select **Save Log File** from the **File** menu. The **Save As** window displays.
3. Navigate to the directory, type a name for your new file, and click **Save**. This operation creates a log file that saves the information displayed in the **Copy** production window.

3.4 Reference—Copy

The topics within this section contain reference information useful while using the application in setup or production.

3.4.1 Windows

The topics within this section provide descriptions of windows accessible from the application. The topics list the user interface element name and include a brief description of actions available from the window.

3.4.1.1 Choose FTP Location

The **Choose FTP Location** window is displayed when you click **FTP** from the **Source** tab or the **Destination** tab. Both these tabs are available in the **Copy Settings** window.

Table 3-1: Choose FTP Location Window

Element	Description
FTP Server	The server name in the format *<ftp.servername.com>*.
Username	Your user name for the selected server.
Password	Your password for access to the server.
Passive FTP Mode	Enables passive mode file transfers between the client and the remote *FTP* site.
Connect	Connects to the server. This displays a list of the directories available on the specified FTP server.

Element	Description
Location	Type a directory name or select a directory from the list displayed when you connect to the server.
Disconnect	Disconnects from the currently active FTP server and enables you to then specify a new FTP server.

3.4.1.2 Copy

The **Copy** window provides access to the Copy module during production. This window is displayed when the Copy module is running in production mode.

Table 3-2: Copy Window

Element	Description
File	Displays the **File** menu.
Run	Displays the **Run** menu.
Help	Provides access to the help system where you can find information to guide you while working in the module.
Run All Batches	Starts processing available batches configured as described in Setting up Run All Batches.
Copy Batch	Displays the **Copy Settings** window for setting batch copying parameters.
Status area	Displays information about the status of the module and tasks in process.

Related Topics

"Copying Batches" on page 34

3.4.1.2.1 File Menu

The **File** menu on the **Copy** window contains the following options:

Table 3-3: Copy window, File Menu

Element	Description
Copy Batch	Displays the **Copy Settings** window.
Save Log File	Displays the **Save As** window. Select a location where the log file should be saved.
Exit	Closes the Copy module.

3.4.1.2.2 Run Menu

The **Run** menu on the **Copy** window contains the following options:

Table 3-4: Copy window, Run Menu

Element	Description
All Batches	Displays the **Copy Settings** window.
Scheduling options	Displays the **Schedules** window.
Stop	Closes the Copy module.

3.4.1.3 Copy Settings

The **Copy Settings** window displays three tabs (Source, Destination, and Options) that assist you in defining the source and destination for files to be copied, as well as log file definition and compression parameters.

The **Copy Settings** window is displayed by clicking **Settings** on the **Copy Setup** window, or clicking **Copy Batch** on the **Copy** window.

3.4.1.3.1 Source Tab

The following settings can be configured on the **Source** tab of the **Copy Settings** window in either Copy Batch or Run All Batches mode during setup. When you select either the **Directory**, **Server**, or the **FTP** option, the window changes settings to accommodate your selection.

Table 3-5: Copy Settings Window, Source Tab

Element	Description
Directory	For retrieving your batch from a local or network directory. This option is not available for Run All Batches mode. • **File name**: type the file name and click the **Browse** button. When retrieving files from a directory, you can use wildcards (***** or **?**) in the **File name** field to specify more than one file at a time. For example, *.ZIP would retrieve all ZIP files in the indicated directory (e.g., `C:\IA\batchfiles\*.zip`). • **Stop task and report error if**: When selected, enables the check boxes for reporting errors. When the check box is selected and the associated error occurs, an error message displays. If cleared, Copy will log an error, skip processing the current file, and continue processing the next file. The available check boxes are: – **No files were found** – **A zip file was unable to decompress** – **A process was not uploaded** – **Unable to activate a process after uploading** – **Unable to delete a zip file**

Element	Description
Server	For specifying the source server when copying your files from a server, or retrieving files from a server. • **Server**: The name of the server where the batch or files currently reside. • **User**: Your user name for the specified server. • **Password**: Your password for the specified server. • **Batch name**: The name of the batch or process. • **Copy only the following stage files**: The **Browse** button displays the **Select Files to Copy** window, where you can select the files to be copied. For more information, see Copying stage files only. • **Copy domain user/group permissions**: Copies domain batch permissions from the source Intelligent Capture server to a destination Intelligent Capture server. For more information, see Copying batch permissions. • **Copy local user/group permissions**: Copies batch permissions that have been granted in local users and groups.
FTP	For retrieving your batch from an FTP site. This option is not available for Run All Batches mode. • **Location**: Specifies the location of the *FTP* server. You can type in the location or file name, or click the **Browse** button to display the **Choose FTP Location** window. If you type the FTP server and file name (e.g. `ftp://ftp.server.com/mybatch.zip`), you can use wildcards (* and ?) in your file names to specify related files. For example, `ftp://ftp.server.com/*.zip` would process all zip files in the specified location. • **User**: Your user name for the FTP server. • **Password**: Your password for the FTP server.

3.4.1.3.2 Destination Tab

The following settings can be configured from the **Destination** tab of the **Copy Settings** window in Copy Batch mode or Run All Batches mode setup. When you select either the **Directory**, **Server**, or **FTP** option, the window changes settings to accommodate your selection.

Table 3-6: Copy Settings window, Destination Tab

Element	Description
Directory	For setting the destination for your batch when copying to a directory. • **File name**: Type the file name or select the file using the browse button. **Note:** Destination zip file names for cannot contain special characters (- _ ' ' ~ # ! @ $ % [ ] { } ( ) +) . • **Insert Value**: Displays the **Choose Value** window for inserting IA Values in file names. • **If file exists**: If the file of the same name exists in the directory, you can specify one of these options for how to handle the situation. • **Overwrite**: Overwrites a file that has the same name. Copy processes the batch and the Copy log displays `Warning: overwrite file already exists`. • **Skip**: Stops processing the current task when a file with the same name exists. Copy moves on to the next available task. The Copy log displays a warning. To finish processing the skipped task, retrigger the batch through Administraton Console. • **Abort**: Stops processing the current batch when a file with the same name as the copied batch exists. • **Prompt**: Displays a prompt when a file with the same name exists. At the prompt, click **Yes** to overwrite the file or **No** to stop processing the current batch.

Element	Description
Server	For specifying the destination server when copying from one server to another one. • **Server**: The name of the server where the batch or files currently reside. • **User**: Your user name for the specified server. • **Password**: Your password for the specified server. • **Batch name**: The name of the batch or process. • **If file exists**: If the file of the same name exists on the destination server, you can specify one of these options for how to handle the situation. • **Overwrite**: Overwrites a file that has the same name. Copy processes the batch and the Copy log displays `Warning: overwrite file already exists`. • **Skip**: Stops processing the current task when a file with the same name exists. Copy moves on to the next available task. The log displays a warning that the task was skipped because a file with the same name already exists. To finish processing the skipped task, the batch can be retriggered through Intelligent Capture Administrator. • **Abort**: Stops processing the current batch when a file with the same name as the copied batch exists. • **Prompt**: Displays a prompt when a file with the same name exists. At the prompt, click **Yes** to overwrite the file or **No** to stop processing the current batch.
FTP	For specifying the FTP server to receive the copied files. • **Location**: Specifies the location of the *FTP* server. Type in the location or file name, or click **Browse** to display the **Choose FTP Location** window. • **User**: Your user name for the FTP server. • **Password**: Your password for the FTP server.

3.4.1.3.3 Options Tab

The following settings can be configured from the **Options** tab of the **Copy Settings** window in Copy Batch mode or Run All Batches mode setup.

Table 3-7: Copy Settings Window, Options Tab

Element	Description
Enable logging	Select this check box to save a copy of the log file. The log file receives statistical information and error messages for processed batches. **Note:** Selecting this option saves the log file for all batches processed through Copy, not just the specified batch.

Element	Description

File path	This field is available when you select the **Enable logging** check box.
	
	Enter the directory path and file name of the log file to save. You can also click the **Browse** button to the right of the field to open the**Save As** window. Navigate to the directory, type a name for the log file, and click **Save**.
	
	This field is available when you select the **Enable logging** check box.
	
	Enter the directory path and file name of the log file to save. You can also click the **Browse** button to the right of the field to open the**Save As** window. Navigate to the directory, type a name for the log file, and click **Save**.
	
	The following variables for the file name are supported:
	
	@(MachineName)
	Represents the machine's `%COMPUTERNAME%` environment variable.
	
	@(PID)
	Represents the current IACopy Windows process ID.
	
	@(Now)
	Represents the machine's current date and time in the form of *<YYYY><MM><DD>- <HH><MM><SS>*.
	
	For example, if the variables are as follows:
	
	`%COMPUTERNAME%` = `scanstation1`
	
	Current date/time = `2014-12-30` at `10:01:23`.
	
	IACopy process ID = `10304`
	
	and the path and file name are as follows:
	
	`c:\logs\iacopy \@(MachineName)_@(Now)_@(PID).txt`
	
	Then the log file path is as follows:
	
	`c:\logs\iacopy \scanstation1_20141230-100123_10304.txt`

Element	Description
Log level	Select the level of logging from this list, from 1 to 5. • Level 1 is the least detailed logging level. It includes all errors and some general information. • Level 5 is the most detailed level and contains all information from lower levels plus debugging information. • Level 1 or 2 should be sufficient for most normal use of the module and provide good economy of log file space. • Levels 4 and 5 are primarily intended for debugging problems and usually are only used when trying to resolve problems. Higher levels will fill the log file more quickly. **Note:** When the log file reaches its maximum size of 5 MB, entries wrap around and begin overwriting the oldest entries in the file.
Convert to JBIG	Select the **Convert to JBIG** check box if you want Copy to compress binary image files in the copied batch using *JBIG* compression. This option does not produce JBIG files, but only uses JBIG compression. This option is useful for saving disk space. **Note:** This check box is only available when copying a batch to a directory or an *FTP* site.
Delete source when finished	Select the **Delete source when finished** check box if you want Copy to delete the original file from the source when copying is complete. To select this option, you must have permission to delete the batch or file. **Note:** This check box is unavailable when copying a file from an FTP site.

3.4.1.4 Copy Setup

The **Copy Setup** window directs your operations during setup. The **Copy Setup** window is displayed when you run Copy for setup.

Table 3-8: Copy Setup Window

Element	Description
Settings	Displays the **Copy Settings** window.
Scheduling	Displays the **Schedule** window. The settings specified here are configured globally for the Intelligent Capture server.
Done	Closes the **Copy Setup** window and saves the settings you selected.
Help	Displays help for the Copy module.

3.4.1.5 Schedule

Use the **Schedule** window to define a new schedule or edit an existing schedule for a department while setting up Run All Batches. To access the **Schedule** window, click the **Add** or **Edit** button from the **Schedules** window.

Notes

- You must specify a schedule if the Copy module step in your process has a department associated with it. If a schedule is not specified, the department will not receive any tasks for the Copy module.
- If the Copy module is run at a time when no departments are scheduled, it processes batches without departments.
- The settings in the Schedule window are configured globally for the Intelligent Capture server. They are not per process and not only for the Copy step in a process. They apply to the entire Intelligent Capture system, and are applicable to any Copy module used in the system.

Table 3-9: Schedule Window

Element	Description
Dept. Name	The department name from the departments specified in the Copy step in your *IPP*.
Time	• **Start**: Sets the time to begin copying queued batches. • **Stop**: Sets the time to end copying.

Element	Description
Frequency	Select a frequency by choosing the appropriate option. • **Every day**: Starts and stops at the specified times every day. • **Selected days**: Starts and stops only on the selected days. Enable the check box next to the day you want to perform the copy. • **Selected dates of each month**: Starts and stops only on the specified date of each month. Enter the numeric date or dates on which the copy will occur.

3.4.1.6 Schedules

The **Schedules** window displays a list of the currently scheduled automatic copy activities. To display the **Schedules** window, click **Scheduling** on the **Copy Setup** window while in Run All Batches setup mode.

Use the **Schedules** window to create schedules, or edit or delete existing schedules.

Table 3-10: Schedules Window

Element	Description
Schedules	Displays information about schedules currently available on the server. Select a schedule from this list prior to clicking the **Edit** or **Delete** buttons.
Add	Displays the **Schedule** window where you can define a new schedule.
Delete	Deletes the currently selected schedule. This button is unavailable if no schedules are selected.
Edit	Displays the **Schedule** window where you can redefine the selected schedule. This button is unavailable if no schedules are selected in the **Schedules** area of the window.

3.4.2 Input IA Values

Copy does not receive input files or produce output files. The following table describes the trigger IA Value used in the`iacopy.mdf`.

Table 3-11: Input IA Values

IA Value	Description
<Ready>	This is the only trigger IA value and must be included in a `Finish` event handler before calling the Copy step. • Attribute: Integer

3.4.3 Output IA Values

Output IA Values include output processing variables, which store data generated by the module during processing, such as the date and time a page was processed. Copy keeps track of the statistics described in the following table.

Table 3-12: Output IA Values

IA Value	Description
<Day>	The month, day, and year the task started and finished, formatted as *<mm/dd/yy>*. • Attribute: String
<Time>	The time the task started and finished formatted as *<hh:mm:ss>*. • Attribute: String

Chapter 4

Multi Module

This section includes the Intelligent Capture utilities module: **ecpcore-cpu**.

4.1 Overview

Multi is a multiple-purpose module. It can manipulate nodes within the batch tree. It can change the effective trigger level in a process. It can also sound a beep to notify when a module finishes processing all tasks for any number of batches. When launched, Multi runs independently of an operator and receives tasks as they become available from the Intelligent Capture Server. Multi receives all of its processing instructions from <*Ready*> trigger values within an *IPP*.

Note: High throughput requirements can require running more than one Multi step.

The Multi module can perform the following functions:

- **Insert nodes in the tree**: Automatically inserts new nodes from levels 1-6 into it. Name the newly inserted node according to data found on a page node by using the IA Values for Multi. The Multi step must be triggered at least one level higher than the highest level node you want to insert.
- **Delete nodes from the tree**: Automatically deletes unnecessary nodes from a batch tree, such as blank pages or pages with barcodes. The Multi step must be triggered at least one level higher than the highest level node you want to delete.
- **Change the trigger level of a module by modifying the process**: Multi can change the effective trigger level of another module. It suspends the processing of the next step until all pages in the current step are processed, preventing the next step from triggering prematurely.
- **Signal end of module processing**: Multi can be programmed to sound a beep after a specified module successfully processes all of its tasks. This works as a signal to an operator that an automated module, such as *OCR*, has finished. The operator then knows to begin manually processing using the next module in the process.

4.2 Setting Up Multi

Most configuration information for Intelligent Capture client modules is defined during module setup. You can run a module in setup mode from Intelligent Capture Designer, from Intelligent Capture Administrator, or from a command prompt with appropriate arguments. Find the details in *OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

When launched in setup mode, a client module displays the **Setup** window in which you can specify the module's configuration settings.

4.2.1 Understanding Triggers

Multi receives all of its processing instructions from <*Ready*> Trigger values within an *IPP*. The <*Ready*> trigger value is an optional variable. It exists for all modules. It triggers any module that is set to any value other than zero. The Multi<*Ready*> trigger value not only triggers Multi but also tells the module which task to perform split, delete, shim, or sound a beep for.

Note: If it receives a task with no trigger set, Multi reports an error. If any step of Multi does not have a trigger set, Multi outputs the error message: `Error: Ready trigger not found`.

4.2.1.1 Using Ready Trigger Values

Declare the <*Ready*> Trigger values you want to use as a global constant in the *IPP*. In the `Finish` event handler of the module step preceding Multi, set the Multi trigger value equal to the <*Ready*> IA Value. For example, if your Multi step is named "Multi", the line <*Multi.Ready = IAMULTI_DELETE*> triggers the module to delete a node.

Set the <*Ready*> trigger value for all nodes of a given level, even for nodes you want Multi to ignore. For example, to delete only a few pages in a node, set the <*Ready*> trigger value on the other pages in that level to <*Multi.Ready = IAMULTI_READY*> or Multi never triggers.

The trigger level of the Multi step must be at least one level higher than the level of the node you want to insert or delete. The <*Ready*> trigger values for the inserted or deleted nodes must be set at the same level as the node. For example, you can delete level 0 nodes (pages) in a level 1 folder node: set <*Ready*> trigger values at level 0 (the page level) to indicate what to do to each page. Then, trigger the Multi`Finish` event handler at level 1 or higher.

Related Topics

"Setting Trigger Levels" on page 61

"Changing the Effective Trigger Level of Another Module" on page 61

"Sounding a Beep When Processing Finishes" on page 62

4.2.1.2 Setting Trigger Levels

The following sample *IPP* code was developed in Process Developer and uses three <*Ready*> trigger values. This sample code splits the tree or inserts a level 1 node before every page with a barcode. It sets the barcode text to be the name of the new node, and then deletes the page containing the barcode.

This sample requires that the following IA Values be declared as global constants so that they can be used throughout the process. You can accomplish this declaration in an IPP by adding the IA Values to the Common Constants module. The *Process Developer Guide* contains information about adding IA Values to the Common Constants module.

```
global const IAMULTI_SPLIT_LEVEL1 = 1
global const IAMULTI_DELETE = 16
global const IAMULTI_READY = 8
```

Related Topics

"Using Ready Trigger Values" on page 60

"Changing the Effective Trigger Level of Another Module" on page 61

"Sounding a Beep When Processing Finishes" on page 62

4.2.1.3 Changing the Effective Trigger Level of Another Module

Multi can change the effective trigger level of another module by inserting a `shim` in your process. The `shim` function suspends the processing of the next step until all pages in the current step are processed, preventing the next step from triggering prematurely.

For example, your process can route page level tasks from ScanPlus to Image Processor. As soon as the Intelligent Capture Server receives a completed task from ScanPlus, it routes this task to a running copy of Image Processor. This routing enables the two modules to work simultaneously on a single batch.

To prevent this premature triggering, modify your process to include a Multi`shim` step triggered at level 7. After ScanPlus finishes processing each page task, the Intelligent Capture Server routes the tasks to Multi. The module holds the tasks until it receives all tasks in the batch, and then all tasks are routed to Image Processor at the same time.

Related Topics

"Setting Trigger Levels" on page 61

"Using Ready Trigger Values" on page 60

4.2.1.4 Sounding a Beep When Processing Finishes

Multi can sound a beep after a specified module successfully processes all of its tasks. Multi sounds a beep on the machine where the module runs as a signal to an operator that an automated module, such as OCR, has finished. The operator then knows to begin manually processing using the next module in the process.

To program Multi to sound a beep after processing finishes, include Multi in an *IPP* using the <*Ready*> Trigger value <*IAMULTI_BEEP*>.

Related Topics

"Understanding Triggers" on page 60

"Using Ready Trigger Values" on page 60

"Setting Trigger Levels" on page 61

4.2.2 Modifying Tree Structures

Using <*Ready*> Trigger values, the Multi module can split the tree by inserting a new node into it, or delete nodes from the tree.

Topics in this section help you configure the Multi module to modify tree structures during processing.

4.2.2.1 Inserting Nodes in the Tree

Multi can split the tree by inserting new nodes from levels 1-6 into it. For example, your scanner does not have barcode reading capabilities so that it cannot recognize a separator page. In this case, use the Image Processor Barcode filter to read the barcode. Then, use Multi to insert a new level 1 node before each page containing a barcode.

You can name the newly inserted node according to data found on a page node by using the IA Values for Multi. For example, you can set the barcode text that signals a split to be the name of the new level 1 node. The *Setting Trigger values* section contains sample code using a split step.

Note: Multi is the only module that can automatically insert nodes.

4.2.2.2 Deleting Nodes from the Tree

Multi can delete any node in the tree. Use this option at any point in your process to delete unnecessary nodes automatically. For example, Multi can delete blank pages or pages with barcodes. Multi can also delete nodes at a higher level.

For example, include separator pages among your documents. Then, have a ScanPlus event or Multi insert a new level 1 node before each separator page. If two separator pages are accidentally scanned in next to each other during scan time, the resulting tree contains two level 1 nodes in a row. The first inserted level 1 node becomes an unneeded document because it contains no child nodes. Multi can delete this unneeded level 1 node.

The following sample code directs Multi to loop through a tree to delete a level 1 node:

```
Option Explicit
Private Sub Split_Finish(ByVal pRoot As IASLib.IAS_RECORD_7)
Dim p1 As IAS_RECORD_1 ' Declare level 1 node
Set p1 = pRoot.Tree.L1Child(0) ' Get the first level 1 node in the batch
While Not p1 Is Nothing  ' Loop through all level 1 nodes
If p1.Tree.NumChildren(0) = 0 Then ' Delete tp1.DeleteEmpty.Ready = IAMULTI_READY
he node if level 1 is empty, else set Multi trigger to ready
p1.DeleteEmpty.Ready = IAMULTI_DELETE
End If
Else
Set p1 = p1.Tree.NextInLevel ' Go to next level 1 node Wend
End Sub
```

The following code provides another way to check for empty nodes and delete them:

```
If p1.Tree.NumChildren(0) > 0 Then_____ 'Document not empty, contains more than one node
p1.Delete_Docs.Ready = IAMULTI_READY
Else
p1.Delete_Docs.Ready = IAMULTI_DELETE___ 'Delete Empty Document
End If
```

Caution

Multi is the only module that can automatically delete nodes. You can delete a node at any level, including the batch level. Do not use Multi to delete a batch immediately after exporting. If you do so, you cannot check the batch if an error displays in the exported files. Instead, use the Timer module to trigger Multi to delete the batch after a specified amount of time, such as a week. Using Timer in this way allows time to check the batch for errors before it is deleted.

Related Topics

"Setting Trigger Levels" on page 61

"Using Ready Trigger Values" on page 60

"Modifying Tree Structures" on page 62

"Ready Trigger Values" on page 66

4.3 Running Multi in Production

After the module has been set up, tested, and placed in a production environment, operators and administrators will run the module in production.

For information related to common production tasks for unattended modules, see *OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

Note: The *Running Modules in Production Mode* section contains production topics that are common to many unattended modules and may not apply to this module.

4.4 Reference—Multi

The topics within this section contain reference information useful while using the application in setup or production.

4.4.1 Windows

The topics within this section provide descriptions of windows accessible from the application. The topics list the user interface element name and include a brief description of actions available from the window.

4.4.1.1 Multi

This table describes the options available in the Multi window.

Table 4-1: Multi window

Element	Description
File menu	Displays the**File** menu for starting, stopping, or exiting the Multi module.
Help menu	Provides access to the *Multi Guide*.

4.4.1.1.1 File Menu

The Multi window **File** menu contains the following options:

Table 4-2: File menu

Element	Description
Start	Starts the Multi module. This option is dimmed when Multi is running.
Stop	Stops the Multi module. This option is dimmed when Multi is not running.
Exit	Closes the **Multi** window.

4.4.2 IA Values

The topics in this section describe IA values that can be used with this application.

4.4.2.1 Input IA Value

Input IA Values include file and processing variables.

Table 4-3: Input IA Value

IA Value	Description
<Title<n>_Input>	You can set this variable to text information read from a page node. For example, in the *IPP*, you can set *<Title0_Input>* to the text of an *OCR* zone and *<Title1_Input>* to the text of a barcode read by the Image Processor barcode filter. Allowed values for *<n>* are from 0 to 9. To add more values for *<n>*, create a custom *MDF* file. • Attribute: String, Input

4.4.2.2 Output IA Values

Output IA Values include file and processing variables.

Table 4-4: Output IA Values

IA Value	Description
<Level<n>_Title<n>>	This string is the title of the newly inserted node and is set to IA Value of *<Title<n>_Input>*. The value of *<n>* is the level of the newly created node (1 through 6). For example, the new level 1 node title can be the text of an *OCR* zone (*<Level1_Title0>*) or the text of a barcode read from a page node (*<Level1_Title1>*). • Attribute: String, Output

Related Topics

"Ready Trigger Values" on page 66

"Input IA Value" on page 65

4.4.2.3 Ready Trigger Values

To include Multi in an *IPP*, use a <*Ready*> trigger value. Define the <*Ready*> trigger values as global constants in your IPP. Then, set the <*Ready*> IA Value equal to the appropriate trigger value. For more information, see "Understanding Triggers" on page 60.

Table 4-5: Ready Trigger Values

<*Ready*> trigger value	Set to:	Description
<*IAMULTI_READY*>	<*8*>	Specified node requires no processing. Multi holds this node until it finishes processing the entire batch.
<*IAMULTI_DELETE*>	<*16*>	Deletes the specified node.
<*IAMULTI_BEEP*>	<*32*>	Sounds a beep on the Multi machine after the specified module finishes processing.
<*IAMULTI_PARENT*>	<*64*>	Performs the indicated operation at the trigger level.
<*IAMULTI_SPLIT_LEVEL1*>	<*1*>	Inserts a level 1 node before a level 0 node.
<*IAMULTI_SPLIT_LEVEL2*>	<*2*>	Inserts a level 2 node before a level 1 node.
<*IAMULTI_SPLIT_LEVEL3*>	<*3*>	Inserts a level 3 node before a level 2 node.
<*IAMULTI_SPLIT_LEVEL4*>	<*4*>	Inserts a level 4 node before a level 3 node.
<*IAMULTI_SPLIT_LEVEL5*>	<*5*>	Inserts a level 5 node before a level 4 node.
<*IAMULTI_SPLIT_LEVEL6*>	<*6*>	Inserts a level 6 node before a level 5 node.

Note: When you use Process Developer to create an IPP, Process Developer adds a Visual Basic module called Common_Constants (`aesconst.bas`) to the project. This module includes declarations of several commonly used constants, which are named values that keep a constant value while the process is executing. If you want to use one of the constants declared in this file, then you can reference it in your IPP. Process Developer includes these constants and functions in a Visual Basic module so that they are available from anywhere within your process. To declare a <*Ready*> trigger values as global constants in an IPP, add them to the Common Constants module. Do not declare them within a module step. The *Process Developer Guide* contains information on the Common Constants module.

Related Topics

"Understanding Triggers" on page 60

"Setting Trigger Levels" on page 61

"Input IA Value" on page 65

"Output IA Values" on page 65

Chapter 5

Timer Module

This section includes the Intelligent Capture utilities module: **ecpcore-cti**.

5.1 Introduction

The Timer module changes IA Values of a batch, or a group of batches, at a user-specified time. During setup, rules are created to specify the conditions under which Timer changes IA Values, and the operations Timer performs during production. These new rules use IA Value strings to locate and change IA Values. After setup, Timer runs in production mode continuously on the network. The module runs in a wait mode until it has to change an IA Value of a batch.

The Timer module includes the following features:

Changes any batch IA Value according to a time set by the user
Timer enables changing the <*Priority*> IA Value of a batch to trigger modules that do not require operator interaction, or changing the <*Ready*> IA Value in the Multi module to trigger Multi to delete batches after a specified amount of time.

Can be configured to run as an unattended service
When run as a service, Timer runs in the background and requires no operator interaction to function.

5.2 Setting Up Timer

Most configuration information for Intelligent Capture client modules is defined during *module setup*. You can run a module in setup mode from Intelligent Capture Designer, from Intelligent Capture Administrator, or from a command prompt with appropriate arguments. For more information, see *OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

When launched in setup mode, a client module displays the **Setup** window in which you can specify the module's configuration settings.

5.2.1 Creating Rules

Timer knows what actions to perform and when to perform them because of rules set in the **Timer Setup** window. Each rule runs once per day at a specified time or continuously throughout the day at specified intervals.

You do not need to include Timer in an *IPP* to use its capabilities. However, you can completely automate actions for Timer with the appropriate IPP and a few rules. When an instance of Timer is included in an IPP, the module accepts tasks, but does not change any IA Values. (Only Timer rules can change IA Values.) Timer receives tasks when the IPP includes a Timer step. If Timer is running, these tasks are processed as soon as they are created. Tasks for Timer are not dependent on a timed rule. For example, if a single rule is set to run at 2:00 AM and a task is created for Timer to run at 6:00 PM, then that task is immediately processed if Timer is running.

Note: To use Timer to run a `Finish event handler` in your IPP, add a step of the Multi module, not the Timer module. You can use Timer to set the <*Ready*> trigger IA Value for Multi to `8`. The `Finish event handler` is called once that step of the Multi module is complete.

5.2.1.1 Setting Up Timer

Use the **Timer Setup** window to configure rules. Access the setup window in setup mode from an *IPP* or in production mode. A Timer rule runs according to the conditions, values, and operation IA Value strings defined on the **Timer Setup** window.

To set up Timer:

1. Run Timer in either setup or production mode. The **Timer** window displays.
2. From the **File** menu, select **Setup**. The **Timer Setup** window displays. The **Rules** text box displays the existing rules. Select a rule description to view its conditions and actions.
3. Select **Edit Protection** to prevent changes to the current rule. To edit the current rule, clear this check box.
4. When you create a rule, type a description in the **Description** field.
5. In the **Time** field, type a value using the <*hhmm*> format to specify the time for the running the rule. For example, type 1700 for 5:00 PM. If the **Time** field is blank, click **Go** to run the rule manually.
6. Select **Run on startup** to run the selected rule when the Timer module starts.
7. Select **Interval (secs)** to indicate that the value in the **Time** field represents an interval length (in seconds) rather than a set time. The interval length is the amount of time that Timer waits before running the rule again.

8. In the **Batch** field, type the batch name affected by the rule. Use the wildcard character (*) to specify groups of batches or all batches. This field is case-sensitive.

9. In the **Condition** field, define a condition which must be true for the Timer module to execute the operation.

   - If the field is left blank, the Timer module always executes the operation at the specified time or interval.
   - If the field contains a condition statement, the Timer module runs the rule when the condition statement is satisfied.

10. In the **Operation** field, define the assignment that occurs when both the **Condition** and **Time** fields are true. Only specify assignment operations in this field. Do not enter arithmetic operations. Operation strings cannot contain spaces.

    The Timer module assigns the number or string to the right of the equal sign to the IA Value defined in the **Operation** field. For example, the **Operation** string `<$batch/priority>=50` indicates that the Timer module has set a value of `50` to the <*Priority*> IA Value of the batch.

11. In the **Value** field, type a value (either a number or a text string) to insert in the **Condition** or **Operation** field. If either of these fields include the <*$value*> placeholder, then the number or string in the **Value** field replaces the <*$value*> placeholder.

    If you specify a value in this field and do not use the <*$value*> placeholder in the **Condition** or **Operation** fields, then the value in the **Value** field has no effect.

12. After creating a rule, click one of the following options:

    - **Add**: Saves the current settings as a new rule.
    - **Delete**: Deletes the selected rule.
    - **Replace**: Replaces the selected rule with the current settings.
    - **Clear**: Clears all **Rule** fields. Does not delete the selected rule.
    - **Go**: Runs the current rule, ignoring the specified **Time** setting.

13. Click **OK** to close the **Timer Setup** window.

Caution

If you select **OK** without selecting **Add**, the **Timer Setup** window closes without saving changes.

14. To exit Timer, click **File** > **Exit**.

5.2.1.1.1 Triggering Modules to Run Overnight

Timer can change the <*Priority*> IA Value of a batch so that unattended modules can run overnight, rather than during the day. The batch priority instructs the Intelligent Capture Server to:

- Process any batch with available tasks without giving precedence to any particular batch (default priority of 50).
- Process the batch before other batches with available tasks (priority of 1-49).
- Process the batch after other batches with available tasks (priority of 51-99).
- Postpone or temporarily halt processing of the batch (priority of 0).

5.2.1.1.2 Deleting Batches at a Specified Time

Timer can change the <*Ready*> IA Value for Multi, triggering a Multi module step to delete a batch after a specified number of days. Deleting exported batches saves space on the Intelligent Capture Server. Waiting to delete the batch until a few days after export provides time to check on any errors received during the export process.

In this sample code, Timer deletes all of the batches three days after exporting the images. The `Export_Finish` event handler sets the batch priority to 0, then sends the batch to Multi to delete. Multi does not run until three days later when Timer sets the priority of the batch to 50.

To delete a batch at a specified time:

1. Include the following sample code in your *IPP*:

```
Option Explicit

Private Sub Export_Finish(ByVal pRoot As IASLib.IAS_RECORD_7)
      ' Initialize the "day" counter for the Timer rules.
   Call IAValueSetLong("$batch=" & pRoot.Tree.BatchID & "/day", 0)
   Call IAValueSetLong("$batch=" & pRoot.Tree.BatchID & "/priority", 0)
   pRoot.Delete.Ready = 16
End Sub
```

2. Start Timer in setup mode, then select **Setup** from the **File** menu.
3. In the **Timer Setup** window, configure the appropriate rules using the following sample as a guide. Substitute the start and stop times and batch names as required.

Table 5-1: Sample Rules for Deleting Batches

Time	Batch	Condition	Operation
1800	*	*<$batch/day>*=2	*<$batch/priority>*=50
1830	*	*<$batch/day>*=1	*<$batch/day>*=2
1900	*	*<$batch/day>*=0	*<$batch/day>*=1

Note: These rules assume that the workday ends at 5:00 PM (1700 hours), which allows an hour for users to log out and for unattended modules to finish processing. If unattended modules need more than an hour to finish the tasks for the day, change the rules to trigger Timer at a later time. Allow enough time for tasks to execute between rules.

Related Topics

"Creating Rules" on page 70

5.2.1.1.3 Triggering an IPP

Use Timer to trigger an *IPP*. In this sample code, a Multi step is triggered, not from within the IPP, but by the Timer module. Timer triggers the Multi step. Multi runs, and the following `Hold_Finish` event handler is called:

To trigger an IPP:

1. Include the following sample code in your IPP.

```
Private Sub Hold_Finish(ByVal pRoot As IASLib.IAS_RECORD_7)
   Dim CompletedDate As Date
   Dim TodaysDate As Date

   CompletedDate = pRoot.Export("ExportCompletedDate")
    TodaysDate = Date
    If TodaysDate - CompletedDate >= 7 Then
       pRoot.DeleteBatch.Ready = 16
    End If

   pRoot.Hold.Ready = 0
End Sub
```

2. Start the Timer module in setup mode, then select **Setup** from the **File** menu.
3. In the **Timer Setup** window, configure appropriate rules using the following sample rules as a guide. Substitute the start and stop times and batch names as needed.

Table 5-2: Timer Example

Time	Batch	Condition	Operation
0200	*	*<$batch/$node>*=1/*<$instance>*=Export/ExportCompleted=TRUE	*<$batch/$node>*=1/*<$instance>*=Hold/ready=8

5.2.1.2 Using IA Value Strings

To locate an IA Value, a Timer module rule uses IA Value strings included during setup. An IA Value string consists of one or more strings indicating the specific IA Value to access, followed by the data contained within the IA Value. Within the Timer module, you can access every IA Value using an *ASCII* string.

Use the following strings to locate an IA Value from a particular step or from a specific node from a specific batch. All these strings are case-sensitive; so for example, *<$Batch>* is not valid.

- *<$batch>*: Value depends on the current task batch number.

  Note: Timer substitutes *<$batch>* for batch names matching the **Batch** field. If a **Condition** or **Operation** string has more than one equal sign (*<$batch/$node>*=1/*<ImageExp.Ready>*=1), then the number or string following the final equal sign is the variable used in the **Condition** or **Operation**.

- *<$instance>*: Value depends on a particular step of a module. Use the string *<$instance>* only with *<$batch>*.
- *<$node>*: Value depends on the current task node. Use the string *<$node>* only with *<$instance>* and *<$batch>*.

The format of the strings (which is similar to a variable assignment statement in many programming languages) consists of two parts:

```
<IA Value> = <data contained within the IA Value>
```

For example, the IA Value string for a step of the ScanPlus module is *<$instance>*=Scan. You can combine strings, such as:

```
<$batch>=250/<$node>=500/<$instance>=Scan
```

5.2.1.2.1 Locating Batch and Node Numbers

Find batch and node numbers in various ways:

- In the Intelligent Capture Administrator, view the batch ID number in the **Batch Traffic** window.
- In the ScanPlus module, configure the module to display node numbers under the thumbnail image of each page in the tree pane. During ScanPlus module setup, at the Page level on the **Levels** tab, add `@N` to the **Display Name** field to display the node number in decimal format.

Note: Batch and node numbers must be typed as decimal values only. Do not enter hexadecimal values.

5.3 Running Timer in Production

After the module has been set up, tested, and placed in a production environment, operators and administrators will run the module in production.

For information related to common production tasks for unattended modules, see *OpenText Intelligent Capture - Module Reference (ECPCORE-CMD)*.

Note: The *Running Modules in Production Mode* section contains production topics that are common to many unattended modules and may not apply to this module.

5.4 Reference—Timer

The topics within this section contain reference information useful while using the application in setup or production.

5.4.1 Windows

The topics within this section provide descriptions of windows accessible from the application. The topics list the user interface element name and include a brief description of actions available from the window.

5.4.1.1 Timer

The following options are available on the **Timer** window:

Table 5-3: Timer Window

Element	Description
File	Valid options are: • **Setup**: Displays the **Timer Setup** window. • **Exit**: Closes the Timer module.
Help	Provides access to the help system where you can find information to guide you while working in the module.
Status pane	Displays the status of the module and the tasks it is processing.

5.4.1.1.1 File Menu

The **File** menu on the **Timer** window contains the following options:

Table 5-4: Timer Window, File Menu

Element	Description
Setup	Opens the **Timer Setup** window where you configure Timer module options.
Exit	Closes the Timer module.

5.4.1.2 Timer Setup

This table describes the setup options available on the **Timer Setup** window. To open this window from the **Timer** window, select **Setup** from the **File** menu. Use the **Timer Setup** window to configure rules for the Timer module.

Table 5-5: Timer Setup Window

Element	Description
Rules	Displays the list of the rules you have entered. Select a rule description to view its conditions and actions.
Edit Protection	Prevents changes to the current rule. All rule settings are disabled until this check box is cleared.
Description	Specifies a description for the current rule. Type a meaningful description in this field to enable users to identify the rule in the future and to differentiate this rule from other rules.
Time	Specifies when the Timer module will run the rule. Type a time in the format <*hhmm*> as digits only (no punctuation). For example, type 1700 for 5:00 PM. • If the **Interval** check box is cleared, the module runs the rule once each day at the specified time. • If the **Interval** check box is selected, the module interprets all digits in the **Time** field as seconds and runs the rule once each time the number of seconds you specified elapses. If the **Time** field is blank, the rule does not run automatically. You can run the rule manually by clicking the **Go!** button on the **Timer Setup** window.

Element	Description
Run on Startup	Runs the selected rule whenever the Timer module starts.
Interval (secs)	Specifies that the **Time** value is to be interpreted as an interval rather than a time of day. • When selected, specifies that the value in the **Time** field is the number of seconds that will elapse before the module runs the selected rule. • When cleared, specifies that the value in the **Time** field is the hour and minute of the day at which the module will run the selected rule.
Batch	Specifies the batch(es) the selected rule will affect. Type the names of batches in this field. You can use the wildcard character (*) to specify groups of batches or all batches.  **Note:** This field is case sensitive.
Condition	Defines a condition which must be true for the Timer module to execute the operation. • If blank, then the Timer module always executes the operation at the specified time or interval. • If it contains a condition statement, the Timer module runs the rule if the condition statement is satisfied. For examples of valid condition statements, see the examples in the topics listed in the Related Topics section below this table.  **Note:** Condition strings must not contain spaces.
Operation	Defines the operation the Timer module executes if the **Time** and **Condition** fields are satisfied. Specify only assignment operations in this field. Do not enter arithmetic operations. For examples of valid operation statements, see the examples in the topics listed in the Related Topics section below this table.  **Note:** Operation strings must not contain spaces.

Element	Description
Value	Specifies a value (either a number or a text string) to insert into either the **Condition** or **Operation** field. If either of these fields include the `$value` placeholder, then the Timer module substitutes the number or string in the **Value** field for the `$value` placeholder when the rule runs. If you specify a value in this field and do not use the `$value` placeholder in your **Condition** or **Operation** field, then the value in the **Value** field has no effect.
Add	Saves the current settings as a new rule.
Delete	Deletes the selected rule.
Replace	Replaces the selected rule with the current settings.
Clear	Clears all Rule fields. Does not delete the selected rule, if any.
Go!	Runs the current rule now regardless of any specified **Time** setting.

Related Topics

"Triggering Modules to Run Overnight" on page 72

"Deleting Batches at a Specified Time" on page 72

"Triggering an IPP" on page 73

5.4.2 Keyboard Shortcuts

The following keyboard shortcuts can be used when running the Timer module.

Table 5-6: Keyboard Shortcuts

Shortcut	Action
ALT+F4	Exit the Timer module.
CTRL+S	Open the **Timer Setup** window.
F1	Display the module Help.

5.4.3 IA Values

The Timer Module Definition File (*MDF*) does not contain any IA Values; however, include the `iatimer.mdf` file in an *IPP* that contains a Timer step. You can add custom variables to this MDF, but it is recommended that you use Dynamic Values in your IPP or create a custom MDF for your variables so they are not overwritten when updating the product.