



P r o f e s s i o n a l E x p e r t i s e D i s t i l l e d

Intelligent Document Capture with Ephesoft

Second Edition

Automate the processing of scanned and digital documents by improving accuracy using web-based open and modern intelligent document capture software

Foreword by John Newton, Founder and CTO, Alfresco Software



Pat Myers
Michael Muller

Ike Kavas
Jon Solove



www.allitebooks.com

Intelligent Document Capture with Ephesoft

Second Edition

Automate the processing of scanned and digital documents by improving accuracy using web-based open and modern intelligent document capture software

Pat Myers

Ike Kavas

Michael Muller

Jon Solove



BIRMINGHAM - MUMBAI

Intelligent Document Capture with Ephesoft

Second Edition

Copyright © 2015 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the authors, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: September 2012

Second edition: August 2015

Production reference: 1200815

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-78355-858-2

www.packtpub.com

Credits

Authors

Pat Myers
Ike Kavas
Michael Muller
Jon Solove

Project Coordinator

Kinjal Bari

Proofreader

Safis Editing

Reviewers

Benjamin Baka
David Brennan
Angelo La Duca
Anita L. Feeley
Megan Hoffman

Indexer

Rekha Nair

Graphics

Sheetal Aute

Production Coordinator

Aparna Bhagat

Acquisition Editor

Vivek Anantharaman

Cover Work

Aparna Bhagat

Content Development Editor

Shali Deeraj

Technical Editor

Vivek Arora

Copy Editors

Tani Kothari
Laxmi Subramanian

Foreword

The world of business has the word *Digital* on its lips like never before. Sure, there have been digital assistants, digital shopping, and digital marketing before, but now, digital is about something more profound. Digital is about businesses transcending an old way of doing business that has been stuck in paper, manual processes, and physical artifacts. Digital is about redesigning your business and the way you work to gain competitive advantages, greater efficiencies, and new sources of revenue.

The four forces of modern technology – the cloud, mobile, social, and big data – have changed the expectations of customers and employees. They cannot expect to do business the same old way. Those who embrace these new forces can totally disrupt entire industries, such as music, media, transportation, and even hotels. Those who do not adapt to these forces go the way of the dodo faster than they can possibly imagine. However, just because you have an established company in the established industry, it doesn't mean that you cannot reinvent yourself if you embrace these forces.

To create a digital business, you must digitize your business processes and information and take full advantage of the speed this information can provide. You can be giving that information to your customers on demand through their mobile devices when they want it. You can be cutting out inefficient paper handling and manual handling to save time and money. You can be freeing up resources such as people, real estate, and manufacturing capacity to create new products and services and add real value for your customers.

Capture is an important step toward moving from a physical world of information handling and manual processes toward a digital world of straight-through processes and customer delight. It can transform expensive urban warehouses full of cabinets of paper into productive, vibrant places of creative energy and collaboration. However, capture is not just scanning a bunch of paper and turning it into the digital equivalent of filing cabinets. Intelligent capture is about extracting information from that paper, discerning meanings, and initiating digital processes to succeed the manual handling it replaces.

Alfresco has worked with Ephesoft for several years to successfully digitize processes in many industries and many companies, big and small. Through open source collaboration and open standards interoperability, Alfresco and Ephesoft have worked hand-in-hand to transform how companies handle contracts, mortgages, insurance claims, accounts payable, and accounts receivable.

The solution provided by Ephesoft with Intelligent Capture, Mobile Capture, Automatic Classification, and the Alfresco with Enterprise Content Management system and its Activiti Business Process Engine simplifies the conversion of these processes into digital business processes. Our joint use of open source also means that these solutions are affordable and that our systems evolve at the speed of community innovation. Proprietary solutions have not done justice to what can be accomplished by truly smart software that embraces the digital world and moves information efficiently and effectively to the delight of your customers and employees.

By picking up this book, I hope that you can realize the benefits of transforming your business process for a new digital era. You can go beyond yesterday's capture solutions of proprietary vendors to a rapid evolution toward a digital future with open source solutions, embracing today's technology forces.

John Newton

Founder and CTO, Alfresco Software

About the Authors

Pat Myers is the executive vice president and a cofounder of Zia Consulting, a content-centric solutions firm. Zia is a platinum Ephesoft and Alfresco partner that provides solutions from paper to mobile. Zia was the Ephesoft Partner of the Year in 2012, 2013, and 2014. Pat has over 14 years of Enterprise Content Management (ECM) experience and 18 years of experience in professional services and application development. Pat and Ike Kavas developed the first official Ephesoft training, and Pat is the coauthor of *Intelligent Document Capture with Ephesoft, First Edition*.

I would like to thank my family, Margaret, Zoe, and Presley, for making my life complete. I would also like to thank them for all their understanding when I am typing away on my laptop. I would also like to thank God for giving me such a wonderful life and surrounding me with terrific people. I would finally like to thank my Zia family for making every day a joy to go to work. Special thanks to Zians Mike Muller, Jon Solove, Nathan McNeel, Anita Feeley, and Megan Hoffman for making this possible.

Ike Kavas has more than 15 years of solid experience in document imaging, document management, workflow, and systems. He is the founder and the chief technology officer at Ephesoft, Inc., responsible for product design and road maps. He has a keen technical background, which he has developed by implementing several multimillion-dollar projects for Fortune 100 companies and has outstanding sales and business experience, which he has demonstrated by achieving and exceeding revenue-based goals.

Kavas is a serial entrepreneur. Before founding Ephesoft, Inc., he managed professional services at Kofax, Inc. and cofounded other technology companies in southern California. He holds a bachelor's of science degree in electronics and electrical engineering as well as a CDIA+ certification.

Michael Muller is the director of engineering at Zia Consulting. He has 25 years of professional software development experience and is currently specializing in enterprise content management.

Jon Solove is a senior solutions engineer and ECM consultant at Zia Consulting, a content-centric solutions firm. He designs, implements, and sells Capture and ECM solutions for a multitude of industries, from manufacturing to financial services. He is a platinum Ephesoft and Alfresco partner who provides solutions from paper to mobile. Zia has been Alfresco's Partner of the Year in 2012, 2013, and 2015 and Ephesoft's Partner of the Year in 2012, 2013, and 2014.

About the Reviewers

Benjamin Baka works as a software developer who considers himself to be language-agnostic and thus seeks out the elegant solutions that his toolset can enable him accomplish. Notable among this language toolsets are C, Java, Python, and Ruby. With a huge interest in algorithms, he always seeks to write code that is, to borrow Dr. Knuth's words, both simple and elegant.

He also enjoys playing the bass guitar and listening to silence.

He currently works with mPedigree Network (a company that hopes to empower consumers, brand owners, and product innovators by building technologies and ecosystems that convert innovation into quality and quality into health, well being, and prosperity) as a technology strategist, weaving together a dizzying array of technologies.

I would like to thank my family for their support and understanding in my endeavors, especially my mum, who exemplifies sacrifice in silence.

David Brennan is a professional services consultant at Ephesoft. He has several years of software development experience and takes pride in customer satisfaction. David takes an interest in learning and working with the latest and greatest technologies. He is also a great problem solver and loves to keep himself busy by working.

I would like to thank to Ike Kavas for giving me the opportunity to work alongside him and his team. I would also like to thank my wife, Moriah, for letting me work those long hours to become a developer.

Angelo La Duca has had a long tenure in Document Imaging and Capture, starting in 1987 as an image processing C programmer before moving into marketing, presales, sales, management, and entrepreneurship.

Before working in IT, Angelo was a photographer and a photography teacher, specializing in B&W fine print. He was also the author of the first Italian book about Zone System.

Angelo has served in multinational firms, with sales and technical responsibilities in Southern Europe, Middle East, and Africa, before cofounding Endurance Italia, the best EMEA partner of Ephesoft in 2013 and 2014. It evolved into Ephesoft Italia in 2015.

I would like to thank my wife, Daniela, for her endless patience – when I say "5 minutes and I will arrive", probably I'm on Pluto's time zone!

A big thanks to the Ephesoft Italia team, Alessandra Zingaretti, Giovanni Cinque, Marco Zuffanelli, and Riccardo Di Mambro, for the great support, fruitful discussions, and constructive and heated confrontations around the most disparate problems we could solve with Ephesoft Enterprise – a lot of new ideas came out of these conversations.

Finally, thanks to Don Field and Ike Kavas – without, you Ephesoft would not have been possible; it's such a great new part of our life!

Anita L. Feeley is a project manager living in Nederland, Colorado. She has over 14 years of experience in software implementations, including project management, business analysis, testing, reporting, and training, as well as database management and XSL stylesheet creation. She has worked in the insurance and financial industries as well as with government agencies. Anita has an MA from the University of Maryland and enjoys reading, biking, hiking, and spending time with her family in the mountains.

Megan Hoffman is a senior project manager at Zia Consulting and has over 13 years of experience implementing and managing software solutions. Throughout her career, she has held a number of positions, including business analyst, tester, trainer, project manager, and product manager.

www.PacktPub.com

Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit www.PacktPub.com.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.PacktPub.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at service@packtpub.com for more details.

At www.PacktPub.com, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.



<https://www2.packtpub.com/books/subscription/packtlib>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read Packt's entire library of books.

Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via a web browser

Free access for Packt account holders

If you have an account with Packt at www.PacktPub.com, you can use this to access PacktLib today and view 9 entirely free books. Simply use your login credentials for immediate access.

Instant updates on new Packt books

Get notified! Find out when new books are published by following [@PacktEnterprise](#) on Twitter or the *Packt Enterprise* Facebook page.

Table of Contents

Preface	v
Chapter 1: A Quick Tour of Ephesoft	1
The user interface	1
Administrative features	2
Batch class management	2
Batch instance management	4
Folder management	5
System configuration	6
Reports	7
The operator user interface	7
Batch list	8
Review validate	9
Web scanner	10
Upload batch	10
File system	11
Summary	12
Chapter 2: Creating a Batch Class	13
Creating a batch class in Ephesoft	14
Creating a document type	16
Training for classification and separation	18
Creating fields	21
Key/value extraction	25
Validation rules	30
Export	32
Copy Batch XML	32
Processing a batch	34
File upload	35
Starting a batch from other sources	37

Review and validation	37
Document review	38
Document validation	39
Summary	40
Chapter 3: Core Ephesoft Features	41
Classification	41
Classification types	41
Search	41
Image	42
Barcodes	42
Automatic	42
One document classification	42
Confidence	42
Search classification	43
Barcode classification	46
Image classification	47
Automatic classification	47
Multiple layouts for a single document type	47
Fuzzy DB	50
Using Web Scanner	55
Uploading batches	57
Export	58
CMIS export	59
Establish a content model in your CMIS	59
Configure the CMIS Export plugin	60
Document type and property mapping	61
Global CMIS configuration	62
Database Export	63
Other export plugins	66
Configuration management of batch classes	66
Summary	70
Chapter 4: Ephesoft's Advanced Features	71
Configuring other classification methods	71
Barcode classification	72
Image classification	73
Fixed form extraction	73
Creating a RecoStar project	75
Configuring the RecoStar project	79
Configuring Ephesoft to use the RecoStar project	81
Table extraction	82
Scripting	88
The workflow scripts	88
Triggering scripts when a field is edited	93

Triggering scripts from a function key	95
The application script	98
The custom workflow	99
Customizing batch class workflows	99
Writing a custom plugin	100
Adding a plugin to Ephesoft	105
The Web Services API	106
Enabling the Web Services API	107
The available web services	107
Invoice processing portal example	108
Creating the upload view	109
Creating the controller and calling the web services	110
Using the app	112
Summary	113
Chapter 5: Tips	115
Troubleshooting	115
Logging	115
Monitoring batch progress	116
Examining the batch file	116
Restarting the batch	117
No blank forms available for training	117
Setting up Active Directory	117
An Overview of LDAP	118
Ephesoft directory server service account	120
Ephesoft directory server groups	121
Ephesoft Active Directory configuration files	121
server.xml	122
user-connectivity.xml	124
application.properties	127
web.xml	127
Active Directory troubleshooting	128
Setting up e-mail processing	128
Summary	130
Appendix: References	131
Glossary	131
Common regular expressions	132
Commonly-modified Ephesoft settings	133
dcma-workflow.properties	133
dcma-batch.properties	133
dcma-cmis.properties	134
Index	135

Preface

In general, document capture refers to the process of scanning paper documents using scanners or cameras, and transforming these documents into an electronic file such as a PDF, TIFF, and so on. Through a type of document capture software, these electronic files are assessed through character or pattern recognition (that is, OCR, ICR, and OMR) and converted into meaningful data or information, also called metadata. The goal of document capture systems is to classify incoming documents into categories and extract metadata by automating processes that humans normally do. By automating this process, organizations can classify documents and route the incoming documents to repositories and workflow systems with all the metadata more efficiently; thus, document capture systems help reduce errors, allow documents to be handled faster, and organizations can scale their business using software rather than labor.

Over the years, the document capture industry has evolved in many ways. Paper documents are still a big portion of the document formats that organizations receive. However, as organizations started to exchange documents electronically, the document capture systems had to evolve to be able to process the documents within e-mail attachments, sent via FTP or by using APIs. Of course, smartphone adoption also influenced the document capture world. As consumers started to produce electronic documents more using their phones, such as check deposits or expense reports, capture using mobile devices became essential.

The capabilities of the capture software has also become more intelligent over the years. Early systems automated the capture process by using barcodes for classifying documents and extracting metadata from predefined areas of structured documents, called forms. Later on, technologies such as Ephesoft classified the documents using several techniques such as document layout or words and phrases that can extract the metadata without defining where the metadata might be located on the document. We call these capture systems *Intelligent Document Capture systems* and they can help organizations automate document capture systems not only for structured documents but also for unstructured documents such as e-mails or documents that do not have any known format. Intelligent Document Capture systems are also easier to deploy and maintain compared to older systems, which provides faster ROI and adoption.

The latest trend in document capture systems is the use of web-based APIs. In the beginning, document capture systems were used only to ingest documents to repositories and workflow systems but with the popularity of APIs and the cloud, a new use case has opened. The newest versions of document capture systems now allow organizations to enable other business applications with document capture functionalities, where the cloud, private or public, based APIs provide document capture services rather than the applications. This makes organizations more efficient when it comes to maintaining, upgrading, and integrating multiple business applications.

If we try to imagine where document capture will be 5 to 10 years from now, it is anyone's guess. However, looking at the emerging technologies and today's advancements, we may be able to predict the future. In the future, we will see machine learning algorithms doing what capture administrators do today, so that any user can simply show the next generation document capture system what he/she wants within that document. Then, the computer should be able to understand what to do the next time without any human intervention. What this means is, if it takes one administrator to configure the system today and one IT professional to set up the servers, in the future there will be no need for either resource. The users will simply request a capture service using a browser or mobile device, the service will learn the human behavior by simply observing the user and then will perform the same task from then on. The projects on which we used to spend months on implementing are now measured in weeks. In the future, they will be measured in hours, if not minutes and seconds.

What this book covers

Chapter 1, A Quick Tour of Ephesoft, takes you on a walkthrough of Ephesoft's user interface. We will look at the administrative and the operator functionality of Ephesoft.

Chapter 2, Creating a Batch Class, explains how to set up Ephesoft. We will see how to create a new batch class and configure it for classification and extraction.

Chapter 3, Core Ephesoft Features, expands on the features introduced in *Chapter 2, Creating a Batch Class*. We will learn more about classification and indexing techniques. We will also learn about web scanning.

Chapter 4, Ephesoft's Advanced Features, explains advanced Ephesoft features. This includes how to set up Microsoft Active Directory, scripting, and utilizing web services.

Chapter 5, Tips, covers the productivity-enhancing tips.

Appendix, References, includes some reference material.

What you need for this book

Ephesoft Enterprise 4.0+ running on a Windows computer.

Who this book is for

This book is intended for information technology professionals interested in installing and configuring Ephesoft for their organization, but it is a valuable resource for anyone interested in learning about document capture in general.

Conventions


In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.


Code words in text are shown as follows: "You can use generic variables, which are `EphesoftBatchID` and `EphesoftDOCID`."

A block of code is set as follows:

```
import com.ephesoft.dcma.da.id.BatchInstanceID;
public interface SamplePluginService {
    void sampleMethod(BatchInstanceID batchInstanceID,
        final String pluginWorkflow) throws Exception;
}
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "Administrators can use the **Up** and **Down** buttons to reorder the plugins or the **Remove** button to remove plugins from the module."

 Warnings or important notes appear in a box like this.

 Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of.

To send us general feedback, simply e-mail feedback@packtpub.com, and mention the book's title in the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide at www.packtpub.com/authors.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files from your account at <http://www.packtpub.com> for all the Packt Publishing books you have purchased. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

Downloading the color images of this book

We also provide you with a PDF file that has color images of the screenshots/diagrams used in this book. The color images will help you better understand the changes in the output. You can download this file from https://www.packtpub.com/sites/default/files/downloads/8582EN_ColoredImages.pdf.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you could report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/submit-errata>, selecting your book, clicking on the **Errata Submission Form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to <https://www.packtpub.com/books/content/support> and enter the name of the book in the search field. The required information will appear under the **Errata** section.

Piracy

Piracy of copyrighted material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at copyright@packtpub.com with a link to the suspected pirated material.

We appreciate your help in protecting our authors and our ability to bring you valuable content.

Questions

If you have a problem with any aspect of this book, you can contact us at questions@packtpub.com, and we will do our best to address the problem.

1

A Quick Tour of Ephesoft

As an introduction to Ephesoft, we will first walk you through the user interface and then examine the installation folder. The locations of certain files and folders within the Ephesoft installation are important because an administrator must make changes here to enable some features.

In this chapter, we will examine the following aspects of Ephesoft:

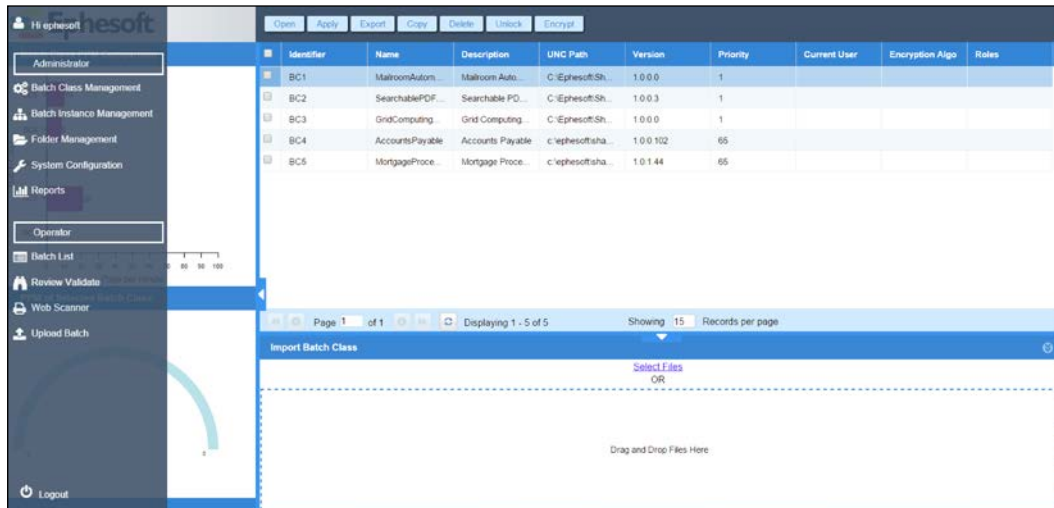
- The user interface
- The installation folder

The user interface

After logging in, users can access Ephesoft's features from an automatically hiding menu of navigation items that we will refer to as the side navigation. To display this menu, simply move your mouse cursor to the left-hand side of the browser window.

Ephesoft has organized this side navigation so that administrative features are separate from the common functions that operators use. Operators typically submit batches and review and validate Ephesoft's output, supplying additional information about the document images being processed.

Administrators enable these activities by defining the operations to be performed on each type of batch. Administrators also monitor and control the processing of the batches.



Ephesoft's navigation menu

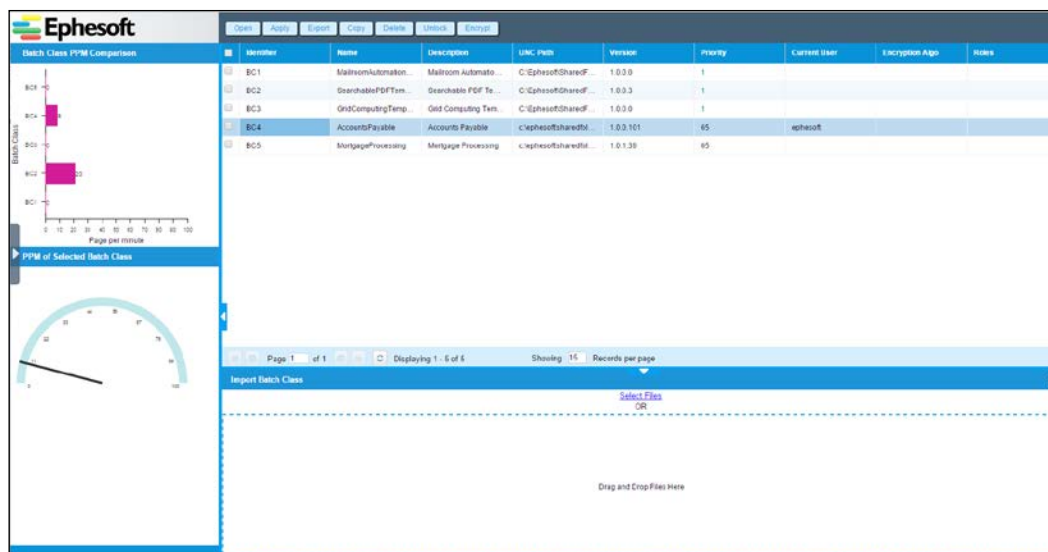
Administrative features

The side navigation provides links to five areas of the system that are commonly used by administrators:

- Batch class management
- Batch instance management
- Folder management
- System configuration
- Reports

Batch class management

A **batch class** defines a set of operations that should be performed on the page images that are provided as input. A batch class consists of document types, document fields, batch class fields, e-mail configuration, and workflow/plugin configuration. The **Batch Class Management** interface allows administrators to create, modify, edit, and delete batch classes.



Ephesoft's batch class management user interface

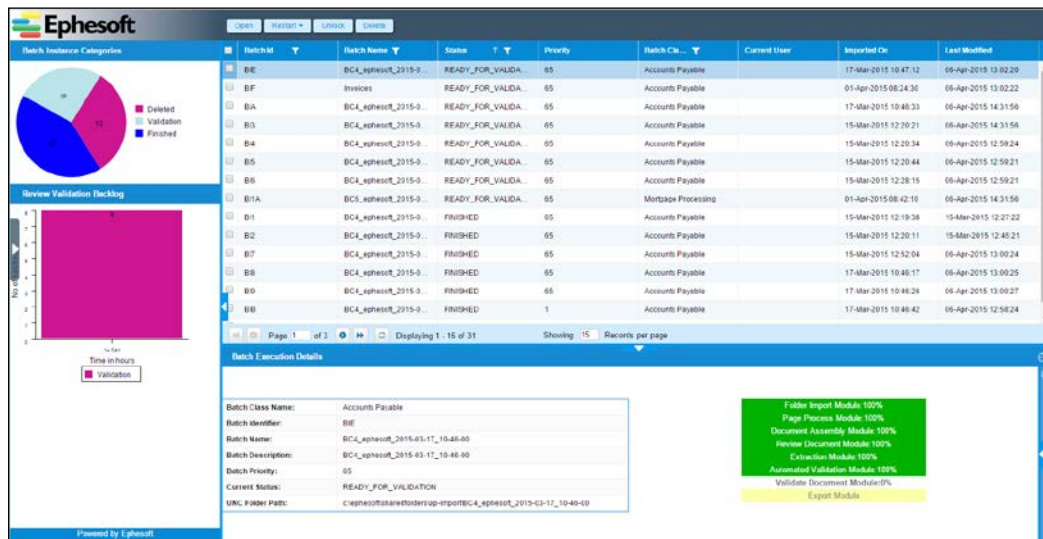
The batch class management interface displays a list of batch classes. Administrators can open a batch class to configure the following:

- **Document types:** The documents that will be processed in the batch class are configured here. Each document type is described by a distinct set of properties called fields. Rules can be configured to extract information from the document into the fields, thereby automating the process of indexing the document.
- **Modules:** Modules are the major steps in the processing of documents. Each module is implemented by a series of plugins.
- **E-mail configuration:** In this portion of the administrative interface, users can provide connection information for an e-mail account, and Ephesoft will process e-mails sent to this address. Ephesoft processes both the e-mail body and the attached documents.
- **Scanner profiles:** This is where administrators can associate one or more scanner configurations with each batch class. These profiles are available in the web scanner.
- **CMIS import:** CMIS is a standard protocol for communicating with document repositories. Ephesoft can use CMIS to monitor a standards-compliant document repository for input.

- **Batch class fields:** Ephesoft can associate information with a batch (the group of page images that are processed together) as a whole. Each piece of information associated with a batch is called a batch class field. Batch class fields are applied to batches and should not be confused with document fields, which contain information that applies to individual documents.

Batch instance management

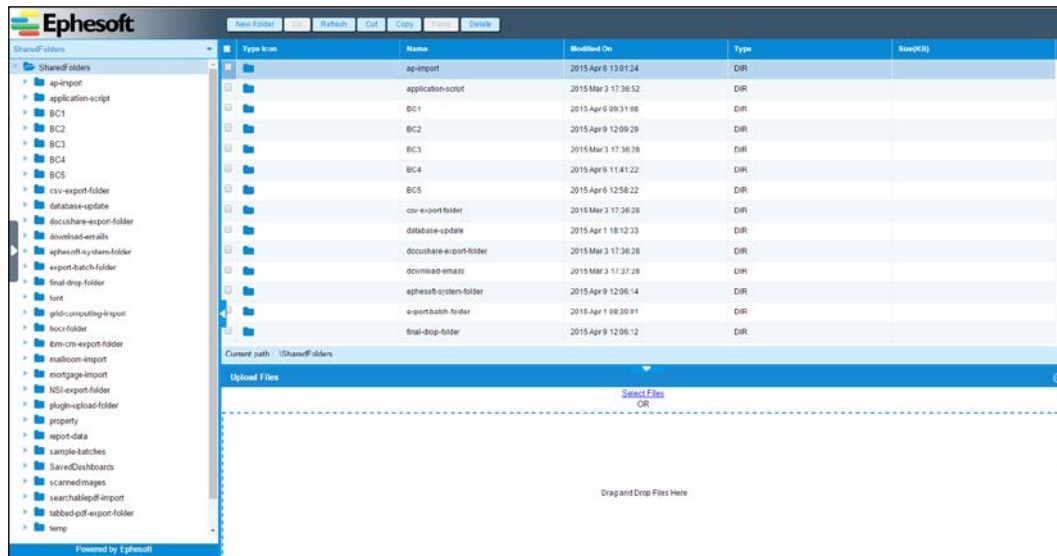
A **batch instance** is a set of page images processed together. The terms batch and batch instance are usually interchangeable. This area within the administrative interface allows administrators to see the status of batches, reprioritize batches, and restart batches in a previous processing step.



Ephesoft's batch instance management user interface

Folder management

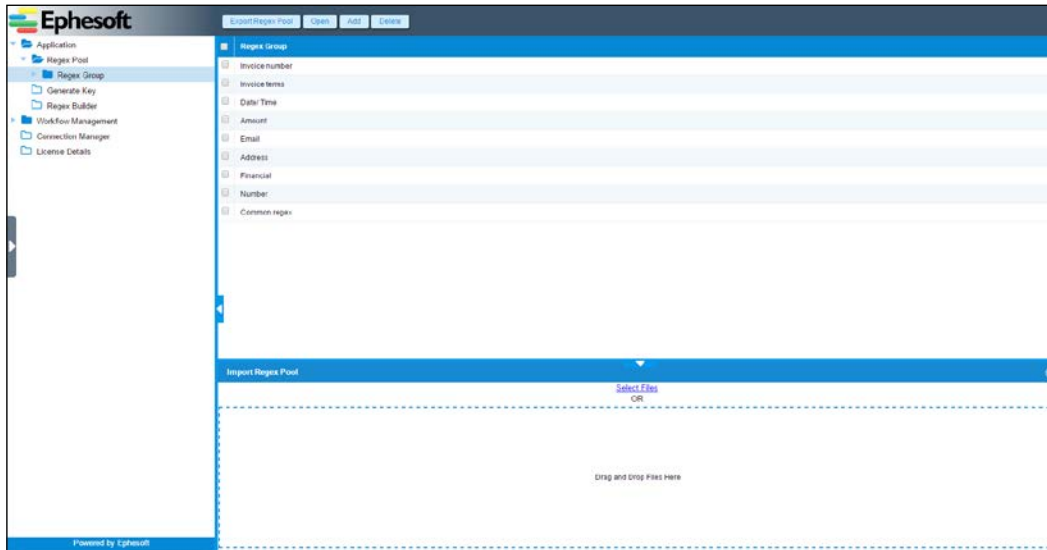
The folder management interface allows the administrator to upload files for batch class configuration. These files are also accessible from the installation folder, but this is often a more convenient way to manipulate these files.



Ephesoft's folder management user interface

System configuration

This administrative interface allows users to manage Ephesoft in ways that are not specific to a batch class or instance.



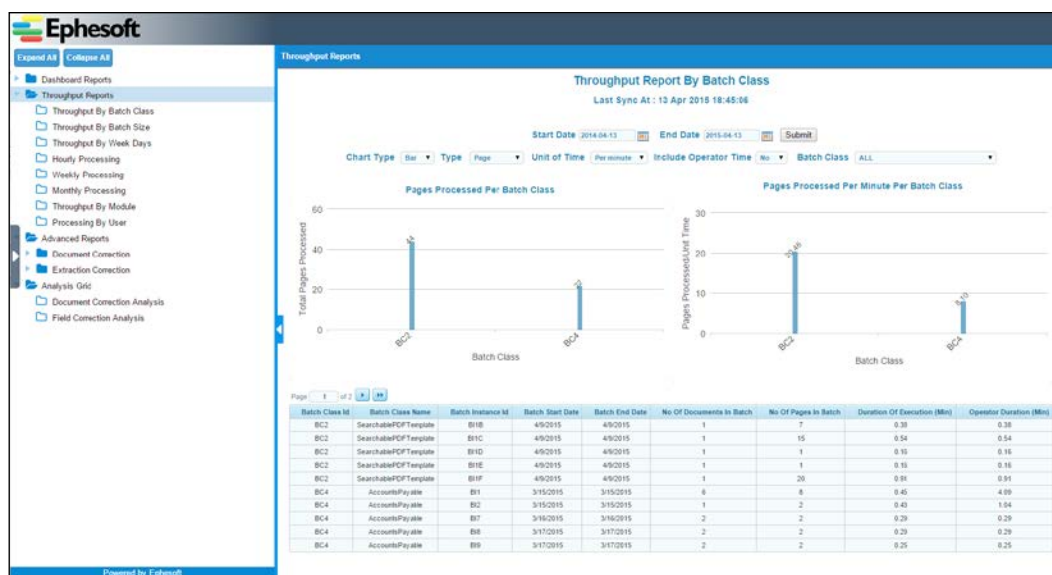
Ephesoft's system configuration user interface

System configuration allows the modification of the following features:

- **Regex pool:** The regular expression pool is a library of regular expressions that administrators can access when creating extraction rules for a batch class.
- **Workflow management:** Ephesoft's features are implemented in components called plugins. The workflow is the sequence in which these plugins are executed. This portion of the user interface allows an administrator to specify what plugins are available when configuring the workflow for a batch class.
- **Connection manager:** The connection manager allows you to create and test database connections. These connections are used by plugins to access databases.
- **License details:** This allows administrators to see the expiration date of the license and the features that are enabled.

Reports

Reporting can be enabled to provide administrators with statistics on the system and throughput. The administrator can filter reports by criteria such as batch class or start date. Advanced reports are also available, including correction reporting. Correction reporting identifies when operators made corrections to Ephesoft's automated processing. This information can be used to optimize the configuration over time.



Ephesoft's reporting user interface

The operator user interface

The side navigation provides links to the following four areas of the system that are commonly used by operators:

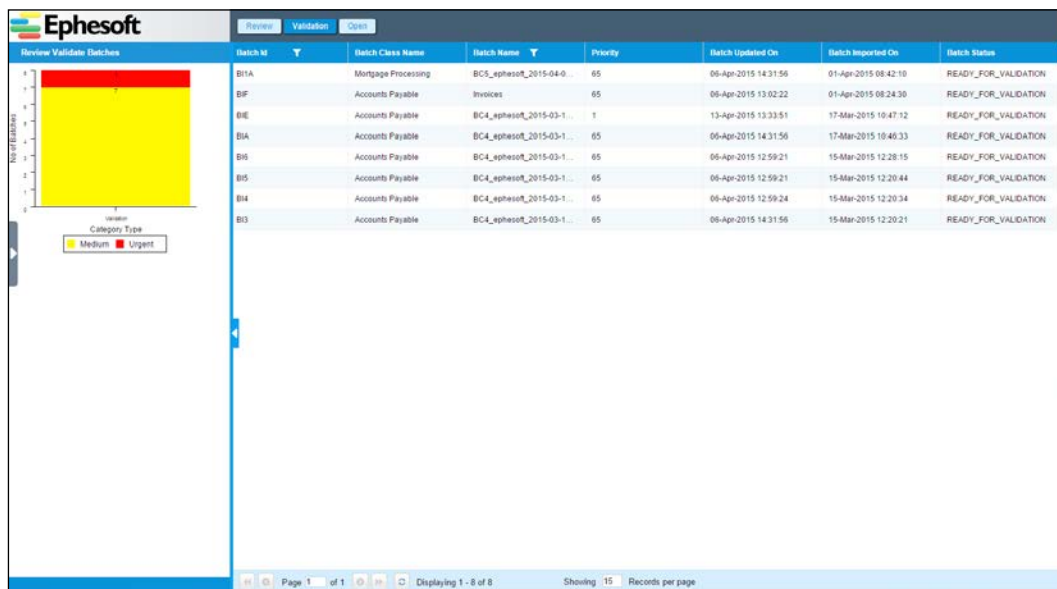
- Batch list
- Review validate
- Web scanner
- Upload batch

Batch list

The **batch list** shows the batch instances that require review or validation.

The review process involves documents that could not be identified as being of a certain type. In Ephesoft, as with most image capture systems, we say that these documents could not be classified. The review interface allows operators to split and merge pages of documents and specify the document type.

The validation process involves fields for which data could not be extracted from the document, or fields where the extracted values do not comply with the previously specified rules.



Review/Validate Batches	Batch ID	Batch Class Name	Batch Name	Priority	Batch Updated On	Batch Imported On	Batch Status
	B1A	Mortgage Processing	BC4_ephesoft_2015-04-0...	65	06-Apr-2015 14:31:56	01-Apr-2015 08:42:10	READY_FOR_VALIDATION
	B1F	Accounts Payable	Invoices	65	06-Apr-2015 13:02:22	01-Apr-2015 08:24:30	READY_FOR_VALIDATION
	B1E	Accounts Payable	BC4_ephesoft_2015-03-1...	1	13-Apr-2015 13:33:51	17-Mar-2015 10:47:12	READY_FOR_VALIDATION
	B1A	Accounts Payable	BC4_ephesoft_2015-03-1...	65	06-Apr-2015 14:31:56	17-Mar-2015 10:40:33	READY_FOR_VALIDATION
	B1G	Accounts Payable	BC4_ephesoft_2015-03-1...	65	06-Apr-2015 12:59:21	15-Mar-2015 12:28:15	READY_FOR_VALIDATION
	B1S	Accounts Payable	BC4_ephesoft_2015-03-1...	65	06-Apr-2015 12:59:21	15-Mar-2015 12:20:44	READY_FOR_VALIDATION
	B1A	Accounts Payable	BC4_ephesoft_2015-03-1...	65	06-Apr-2015 12:59:24	15-Mar-2015 12:20:34	READY_FOR_VALIDATION
	B1G	Accounts Payable	BC4_ephesoft_2015-03-1...	65	06-Apr-2015 14:31:56	15-Mar-2015 12:20:21	READY_FOR_VALIDATION

Ephesoft's batch list user interface

Review validate

The review validate screen will present the operator with the next available batch for processing according to priority and batch date.

The screenshot displays the Ephesoft review validate interface. On the left, a list of documents (DOC1, DOC2) is shown. The main area displays a batch of invoices with the following details:

- Batch ID: BIE
- Batch Name: BCL_sphen...
- Priority: 65
- Document Type: Invoice
- Header Info:
 - Invoice Number: 68193495
 - Invoice Date: 03/25/08
 - Vendor Name: ACSC
 - Vendor Number: 88888
 - Total Amount: 130.64

On the right, a detailed view of an automobile insurance billing statement is shown. The statement is from the Interinsurance Exchange of the Automobile Club (IAC) and is for a policy number 68193495. The billing date is 3/25/08, and the due date is 4/10/08. The statement includes a table of charges and a total amount due of \$20.00.

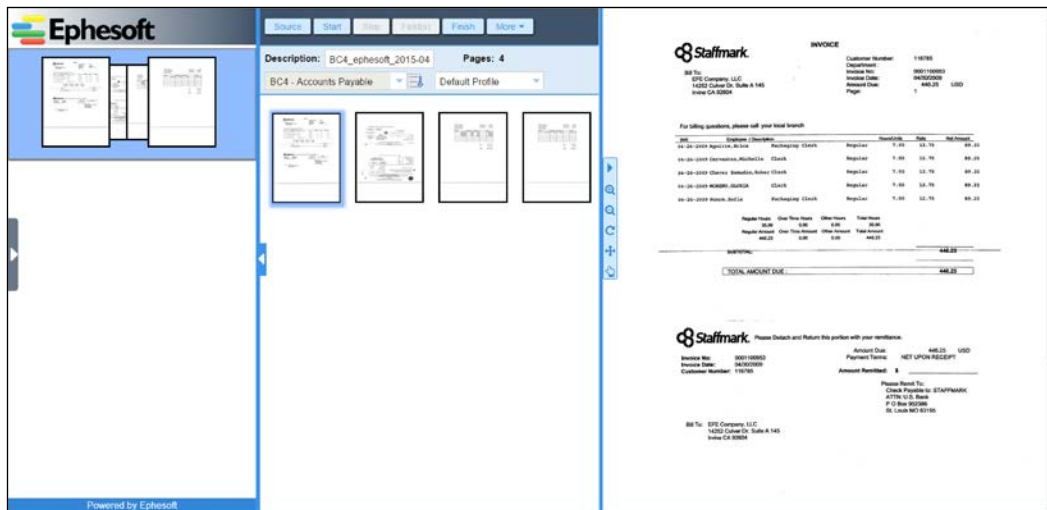
1.	2.	3.	4.	5.	6.	7.	8.
Previous Balance	\$ 130.64					New Balance	\$ 20.00
Credits	\$.00					Minimum Due	\$ 20.00
Payment	\$ 130.64						
Unpaid Balance	\$.00						
Finance Charge	\$.00						
Additional Premium	\$ 20.00						

The statement also includes a QR code and a note: "THANK YOU FOR CHOOSING THE AUTO CLUB FOR YOUR INSURANCE NEEDS".

Ephesoft's review validate user interface

Web scanner

Ephesoft is capable of capturing content from a scanner attached to the user's workstation. What is unique about the web scanner is that no software needs to be installed on the workstation; Ephesoft uses a Java applet to send content directly to the server from any TWAIN-enabled scanner.

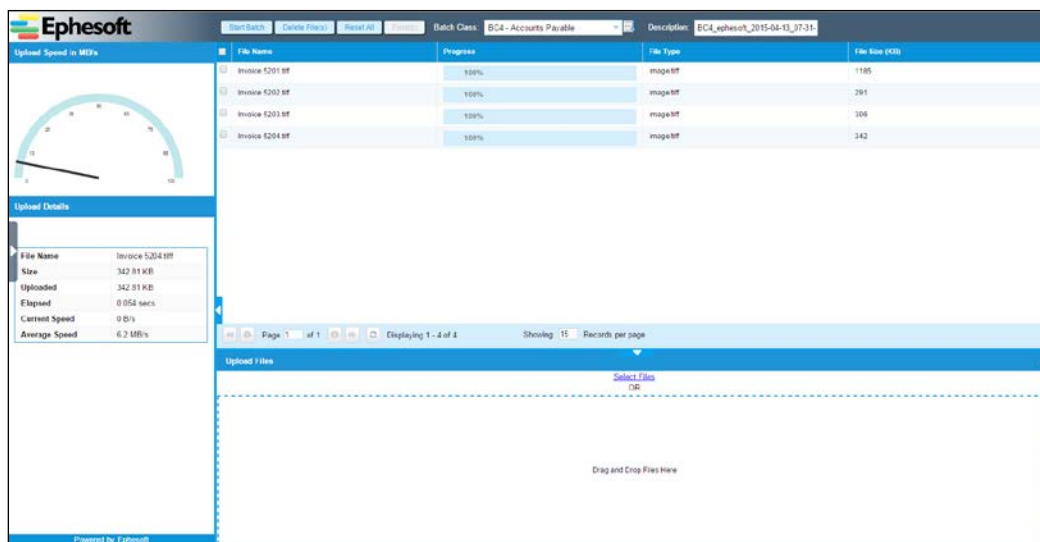


Ephesoft's web scanner user interface

The first time a user logs into the operator interface and selects the **Web Scanner** link on the side navigation, the user will have to choose a scanner. When the user selects the **Source** button, the user will be shown all TWAIN devices that have been installed on the user's workstation. Once the scanner is selected, the user can select the batch class to be used for processing and start the scan job.

Upload batch

Operators can submit PDF and TIF files directly to Ephesoft for processing by using the upload batch feature. Once the documents are selected and uploaded, the operator can select the appropriate batch class and start the batch processing.



Ephesoft's upload batch user interface

File system

The following are some important folders that are created when Ephesoft is installed. These are subfolders beneath the Ephesoft installation folder:

- **Apache 2.2:** Apache can be used in front of Ephesoft for load balancing and failover. It is included in the installation but not configured.
- **Application:** The Ephesoft Java web application is installed in this folder.
- **Application/i18n, themes:** These folders contain files to customize and localize the Ephesoft application.
- **Application/native/RecostarPlugin:** This plugin provides the image OCR functionality.
- **Application/WEB-INF/classes/META-INF:** System configuration property files are stored in this folder.
- **Dependencies/gs, ImageMagick:** Applications that Ephesoft uses for image manipulation are installed here.
- **Dependencies/licence-util, licensing:** These folders contain tools to collect the information needed to generate and install license keys.
- **Dependencies/luke:** Luke is a tool that helps troubleshoot problems with Lucene indexes.

- **JavaAppServer**: This folder contains the Tomcat configuration for Ephesoft.
- **JavaAppServer/conf**: This is where the contexts are defined for Ephesoft; it is where URLs are bound to java code.
- **EphesoftReports**: The configuration and binaries for reporting are stored here.
- **SharedFolders/BC99**: The configuration for each batch class is stored here. The contents of the batch class folder can be modified through the **Folder Management** interface by a batch class or system administrator.

Summary

In this chapter, we looked at the administrative and the operator functionality of Ephesoft. We also looked at the installation folder on the filesystem. It's time to put Ephesoft to work.

In the next chapter, you'll learn how to train the system to recognize your documents, extract content from them, and test the configuration.

2

Creating a Batch Class

A **batch** is a set of pages or page images to be processed. A **batch class** is the definition of how Ephesoft will process these pages. A **batch instance** is the actual process of performing the operations defined by the batch class. Each batch class can monitor different sources for incoming content and create a batch instance when the content arrives. Batch classes can monitor a variety of sources, including folders in the file system, e-mail accounts, and standards-compliant content repositories.

As an example, we are going to create a new batch class for an accounts payable group that wants to automate the processing of invoices.

As we work through the example in this chapter, we will cover the following topics:

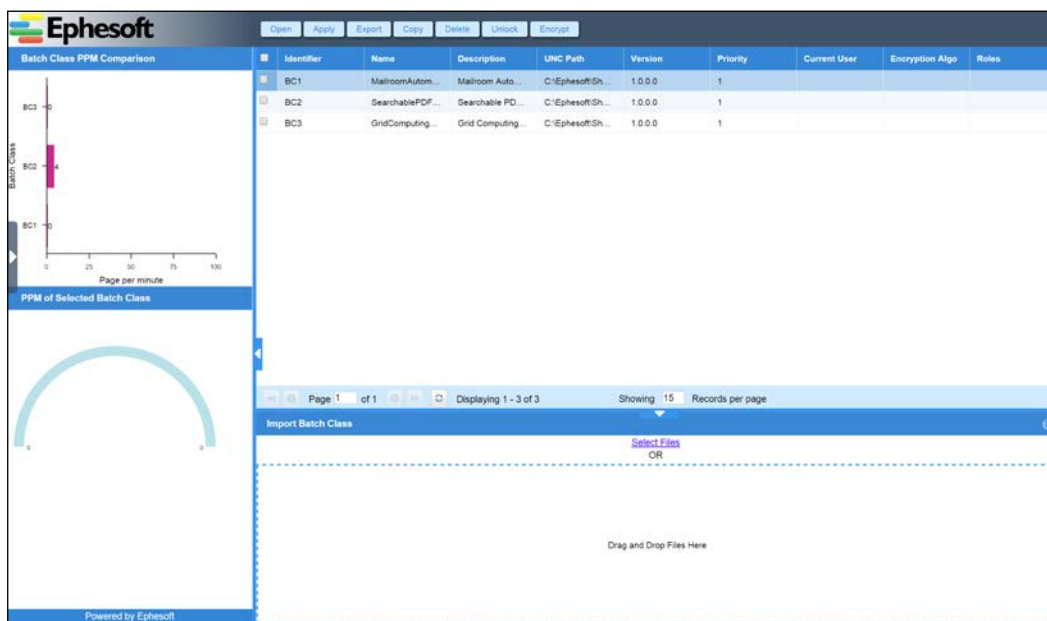
- Creating a batch class
- Creating a document type
- Training for classification and separation
- Creating fields
- Basic key/value extraction
- Validation rules
- Exporting
- Processing a batch

Creating a batch class in Ephesoft

The first step in our example is to create a batch class for processing accounts payable documents. We will accomplish this by copying a batch class provided by Ephesoft, and then customizing this batch class as necessary.

The batch classes that come with Ephesoft provide the core behavior that most organizations need. These batch classes should be used as templates and should not be modified or deleted. Creating a copy maintains the original template batch class for use in creating future batch classes.

We will create our batch class on the basis of the `MailroomAutomationTemplate` batch class. This batch class comes preconfigured with most Ephesoft functionalities, although some of it is disabled. To copy the `MailroomAutomationTemplate` batch class, first select it from the list in the Batch Class Management interface, and then click on the **Copy** button above the list.



The batch class management main screen

Enter the **Name**, **Description**, **Priority**, and **UNC Folder** for the new batch class. **Priority** can be a value from 1 to 100, with 1 being the most urgent. Instances of this batch class will inherit this priority. Processing of higher-priority instances will take precedence over that of lower-priority instances. The Universal Naming Convention folder can be a UNC path (like \\server\remote\path) or local path (like c:\local\path). Ephesoft will monitor this folder for content such as scanned page images. When content is found, Ephesoft will create a new batch instance, performing this batch class's operations on the page images. After the information is entered, click on the **OK** button:

GridComputingTemplate Grid Computing Template C:\Ephesoft\SharedFolder...

Copy Batch Class

Name:* AccountsPayable

Description:* Accounts Payable

Priority:* 65

UNC Folder:* \\sharedFolders\ap-import

OK Cancel

New batch class properties

The new batch class will appear at the end of the list of batch classes in the **Batch Class Management** interface:

Ephesoft

Open Refresh Export Copy Delete Unlink Erase

Identifier	Name	Description	UNC Path	Version	Priority	Current User	Encryption Algo	Rules
BC1	MailroomAutom...	Mailroom Auto...	C:\Ephesoft\Sh...	1.0.0.0	1			
BC2	SearchablePDF...	Searchable PD...	C:\Ephesoft\Sh...	1.0.0.0	1			
BC3	GridComputing...	Grid Computing...	C:\Ephesoft\Sh...	1.0.0.0	1			
BC4	AccountsPayable	Accounts Payable	C:\Ephesoft\Sh...	1.0.0.44	65			

Page 1 of 1 Displaying 1 - 3 of 3 Showing 15 Records per page

Import Batch Class

Select Files
OR

Drag and Drop Files Here

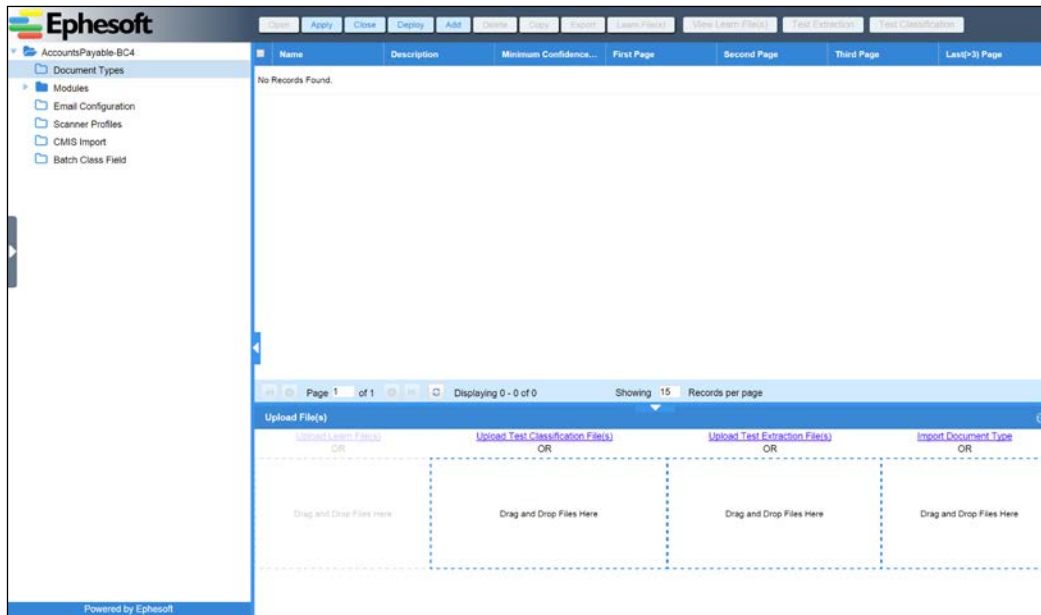
The new batch class in batch class management

Creating a document type

Now that the accounts payable batch class exists, we must tell Ephesoft about the types of documents that this batch class will process. Each batch class can process many different document types, but we will begin by defining just one document type: an invoice.

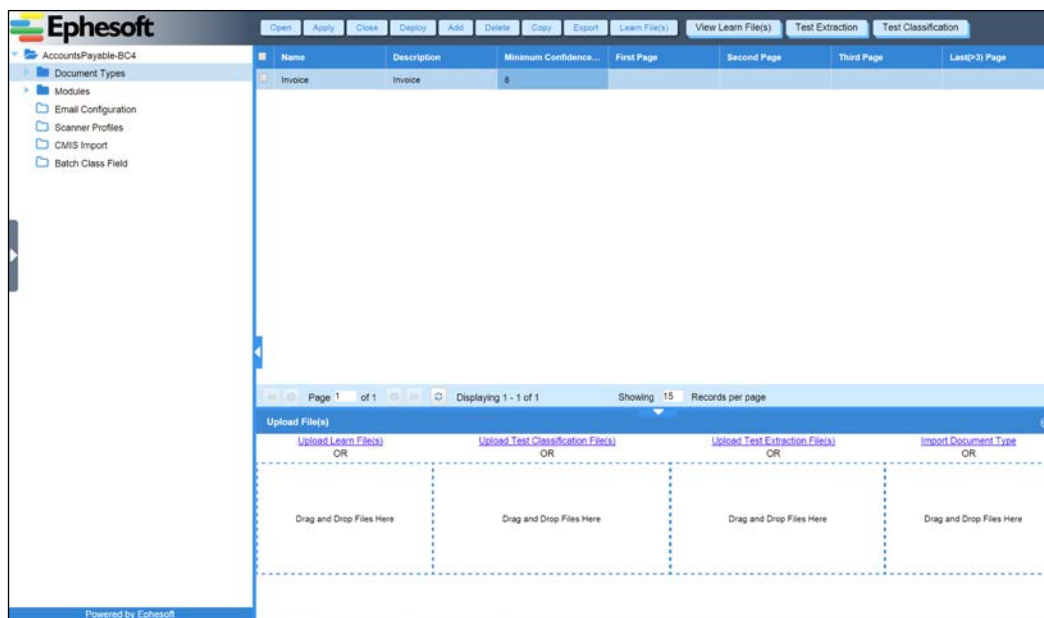
Select the newly created batch class and click on the **Open** button.

When editing batch classes, Ephesoft displays a navigation tree on the left that allows the user to select the aspect of the configuration to be modified. The **Document Types** item is the default, so it should already be selected, as shown in the following screenshot:



The batch class document type list


Click on the **Add** button located on the top of the screen. A new row will appear in the **Document Types** table into which you can enter information about the new document type, as shown in the following screenshot:



New document type configuration

The following information can be provided when creating a new document type:


- **Name:** The name of the new document type. Use a short name here, such as Form1040, Invoice, or Application.
- **Description:** This field is used to provide additional identifying information to operators, such as U.S. Individual Income Tax Return.
- **Minimum Confidence Threshold:** This must be a value between 0 and 100. When Ephesoft determines the document type, Ephesoft calculates the confidence score. If the calculated confidence is below the document's minimum confidence threshold, then an operator will have to review Ephesoft's choice for the document type.
- **First Page, Second Page, Third Page, and Last (>3) Page** (if applicable): These fields allow you to specify a configuration file for use by the OCR engine used by Ephesoft. This configuration is page-specific, as opposed to document-specific. You may want, for example, to have the OCR engine look for a signature at a specific location on the last page of a certain document type. To accomplish this, create a configuration file to look for the signature, and then associate this configuration file with the last page of this document type. This sort of low-level OCR engine configuration is necessary for advanced extraction of content such as handwritten text and check boxes. This is covered in greater detail in the next chapter.

 The OCR settings for the last page will only be applied to documents that are more than three pages long.

The value that you choose for the document's minimum confidence threshold can have a large impact on the efficiency of your system. If you enter a value that is too low, then documents could be incorrectly identified as the wrong type. If you enter a value that is too high, operators will have to waste time reviewing and confirming the correctly identified document types.

It is important to understand that the confidence values are scores and not percentages. Suppose Ephesoft says that it's confident that a document is of the invoice type 12. This does not imply that the document will be an invoice 12 percent of the time, or anything of the sort. It is just a score, on a scale of 1 to 100.

For our example, enter *Invoice* for both the name and description, use an arbitrary threshold value of 8 (we will discuss confidence and threshold configuration in the next chapter), leave the other fields blank, and then click on the **Apply** button.

 When confirmation dialogs appear, such as when you delete a document type, you can press the *Return/Enter* key, rather than clicking on the **OK** button with your mouse.

When you make any working change, such as deleting a set of document types or a working extraction rule, click on the **Apply** button to save the changes. You can also click on **Save**, but this will redirect you back to the batch class management home page. If you navigate away from a page without saving or applying your work, your changes will be lost.

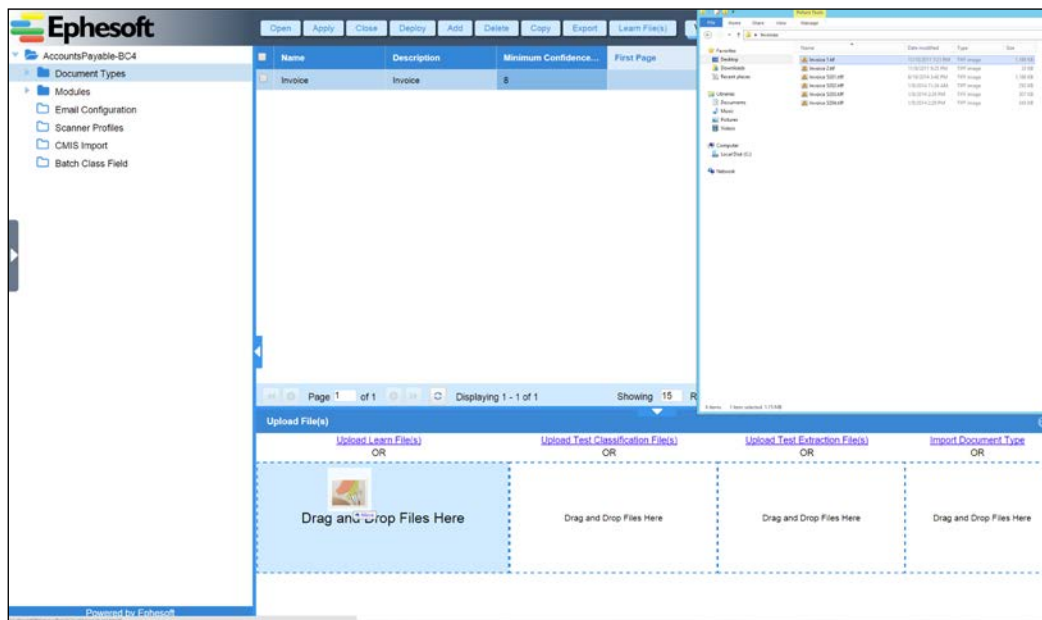
Training for classification and separation

Traditional image capture systems require that scan operators place separator sheets between each document in a batch. The separator sheets tell the system where each document begins and ends and also indicate the document type. We will configure Ephesoft to separate the example accounts payable batch into individual documents and classify these documents as being invoices without the use of separator sheets. In order to accomplish this, we must train Ephesoft to recognize invoices by supplying an example.

First, find or create a blank invoice. If a blank invoice is not available, you can use an actual invoice with line items and header information, but this won't be as effective. We will discuss training and sample documents further in the next chapter.

Select the checkbox next to the invoice document type. Then, drag and drop the sample invoice to the area below the **Upload Learn File(s)** link.

Ephesoft will display a message while it learns the sample invoice you uploaded. The learning process involves discovering and recording the text that is on the first page, the last page, and the middle pages of the sample.

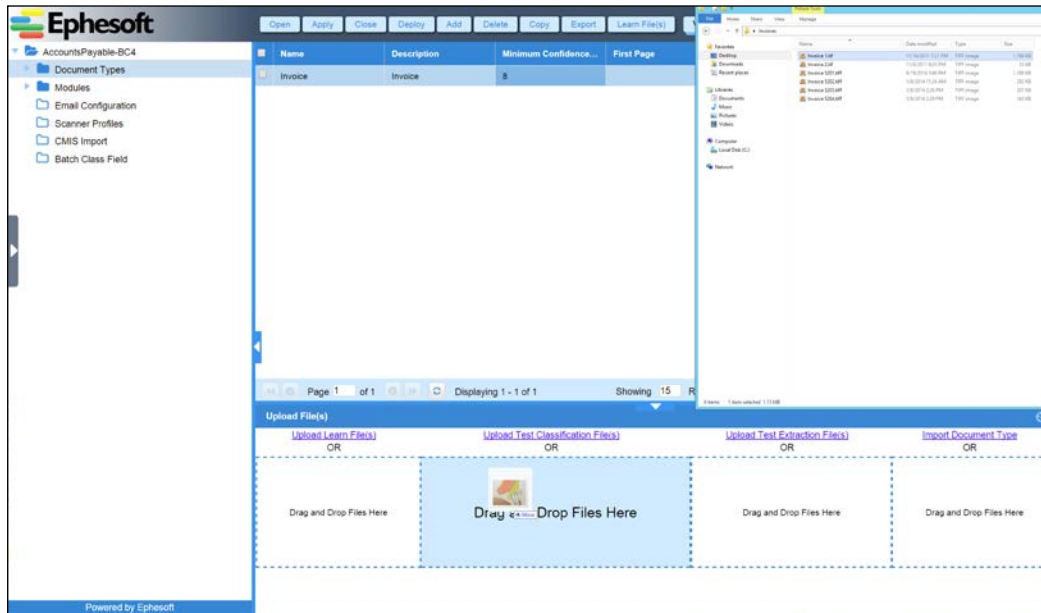


Drag and drop classification samples

Ephesoft will notify you when the learning process is complete.

Creating a Batch Class

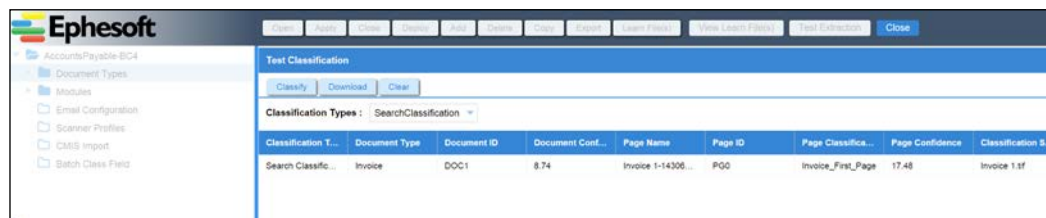
Next, test the classification of invoices. Find an invoice (an actual invoice, not a blank template), and drag and drop it into the area below the **Upload Test Classification File(s)** link.



Drag and drop classification samples for testing

When you upload the test classification files, Ephesoft does not immediately begin testing the classification. After you receive notification that the upload is complete, click on the **Test Classification** button, and then click on the **Classify** button. Ephesoft will look at the text in the test files and compare that to the text in the files that it learned as being an invoice. The document type that has sample pages with text that best matches the test invoice that you uploaded will be selected as the document type for your sample.

The results of the classification test are displayed in a table.



The screenshot shows the Ephesoft Test Classification window. On the left is a sidebar with a tree view containing 'AccountsPayable-BIC4', 'Document Types', 'Modules', 'Email Configuration', 'Scanner Profiles', 'CMS Import', and 'Batch Class Field'. The main area is titled 'Test Classification' and contains a 'Classify' button, a 'Download' button, and a 'Clear' button. Below these is a 'Classification Types' dropdown menu set to 'SearchClassification'. A table displays the results with the following columns: Classification T..., Document Type, Document ID, Document Conf..., Page Name, Page ID, Page Classifica..., Page Confidence, and Classification S... The table contains one row with the following data: Search Classif..., Invoice, DOC1, 8.74, Invoice 1-14306..., PGO, Invoice_First_Page, 17.48, and Invoice 1.tif.

Classification T...	Document Type	Document ID	Document Conf...	Page Name	Page ID	Page Classifica...	Page Confidence	Classification S...
Search Classif...	Invoice	DOC1	8.74	Invoice 1-14306...	PGO	Invoice_First_Page	17.48	Invoice 1.tif

The test classification results

The most important columns to look at are as follows:

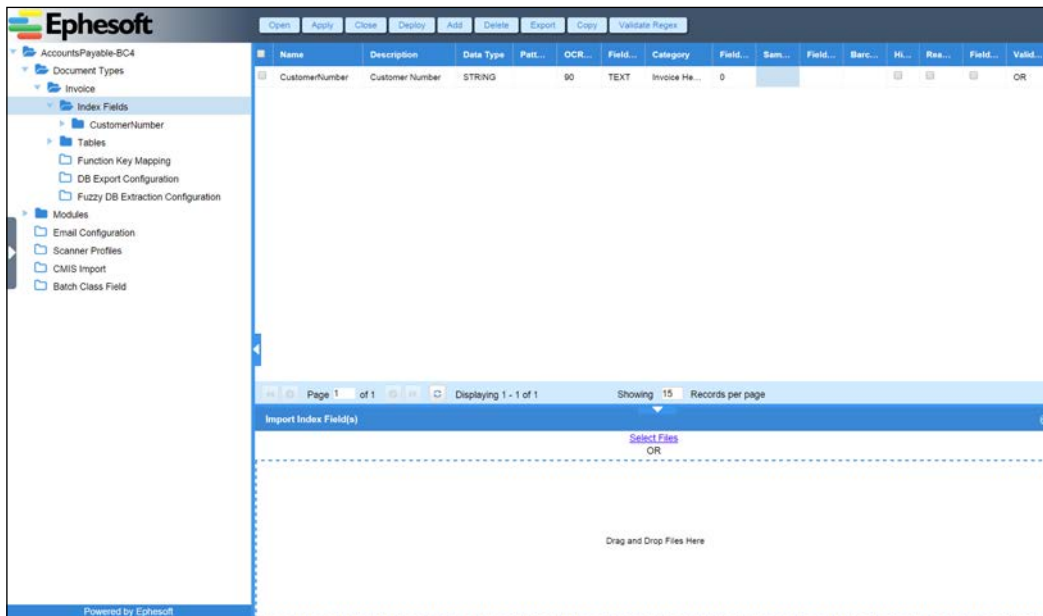
- **Document Type:** This is the document type that Ephesoft assigns to the test page.
- **Document Confidence:** This score indicates the certainty that Ephesoft has in its decision for the document type assignment. Remember that this number does not need to be large in order for Ephesoft to work effectively, it just needs to be larger than the score associated with the other document types that you trained for this batch class.
- **Document ID:** Use this column to determine whether Ephesoft is correctly identifying document boundaries. All pages belonging to the same document will share a common document identifier. Click on the **Close** button to return to batch class management for the accounts payable batch class.

Creating fields

Ephesoft stores information about a document in document-level fields. Further, Ephesoft stores information common to all the documents in a batch in batch-level fields. Typically, field content is extracted from a document, but the fields can be populated in a number of other ways. The field content can be manually entered by an operator, for instance, or populated from a database, or calculated programmatically on the basis of the values of other fields.

Creating a Batch Class

For the invoice document type, we would like to extract the customer number, invoice number, and invoice date. In order to accomplish this, we must first create the fields in the document type that we created. From the **Batch Class Management** interface, select and edit the **Accounts Payable** batch class. From the **Document Types** tree node, open the **Invoice** document type. The **Index Fields** item should be selected by default. Click on the **Add** button at the top of the user interface, which will create a new row in the table of document-level index fields.



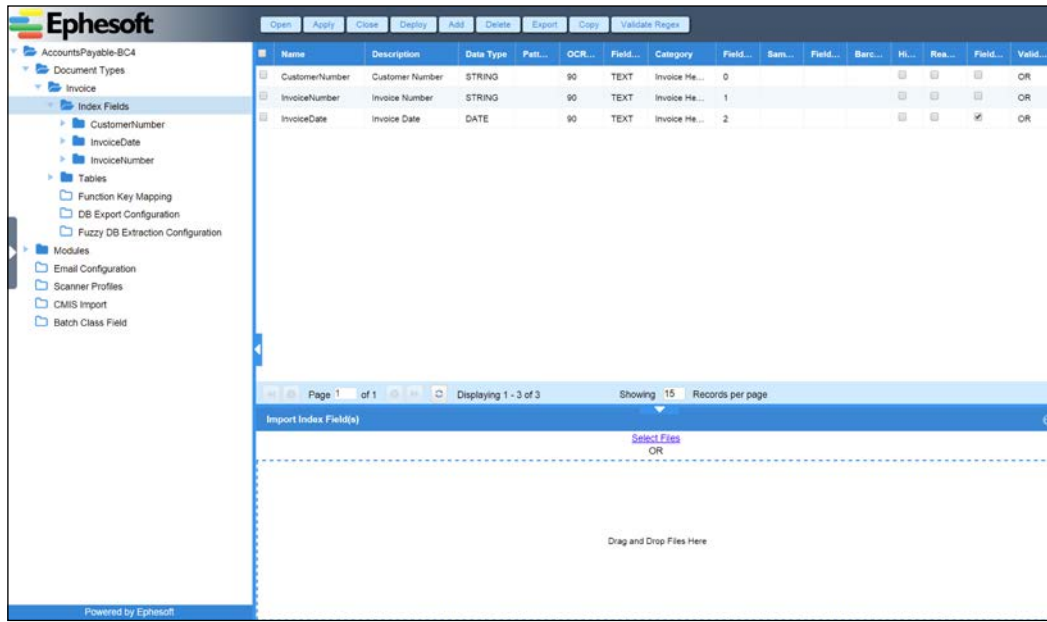
The index field configuration

Ephesoft will allow the following information to be edited in this row:

- **Name:** Name of the field.
- **Description:** Description of the field. This is what is displayed to the operator.
- **Data Type:** Options include string, date, long, double, integer, float, big decimal, Boolean, and password.

- **Pattern:** Ephesoft will search the entire document for words that match this regular expression. This is useful when the data to be extracted has a distinctive format (like a social security number), but may occur anywhere in the document. Ephesoft requires you to check the syntax of regular expressions by clicking on the **Validate Regex** button above the table.
- **OCR Confidence Threshold:** This is the threshold of the OCR confidence for the field. If the extracted OCR confidence is below this threshold, it will be marked in the validation for operator review.
- **Field Type:** This specifies the type of widget that Ephesoft should use when displaying this field in the validation user interface. Options include text, list, checkbox, date, multiline, and combo. For instance, the date option will give the users in validation a calendar selection widget.
- **Category:** This specifies a grouping of fields for the validation user interface
- **Field Order:** This is a number that Ephesoft uses to determine the order in which the fields are presented in the operator's interface.
- **Sample Value:** This text is displayed in the operator's interface as a hint, indicating the valid format for data entry. A date field might say MM/DD/YYYY, for instance.
- **Field Option Value List:** This allows the administrator to create a drop list of values from which the operator can select. A semicolon is used to delimit the values.
- **Barcode Type:** It is possible to encode field values in a barcode. This specifies what type of barcode is used to encode the value for this field.
- **Hidden:** Some fields may be used internally and should never be presented to the operator. The administrator may indicate that such fields are hidden.
- **Read Only:** Some fields may be viewable but not editable by the operator. The administrator may indicate such fields are read only.
- **Field Value Change Script:** This flag indicates that edits to this field should trigger a script.

- **Validation Operator:** This controls how validation rules are applied. Fields with **AND** will have to comply with all of the fields' validation rules. Fields with **OR** will have to comply with at least one.



The index fields for invoice document type

In the preceding example, we have added fields so that the following information can be associated with each invoice in the Accounts Payable batch class:

- Customer number
- Invoice number
- Invoice date

Once all fields are created, click on the **Apply** button to save the progress.

Key/value extraction

Extraction is the process of automatically populating fields with text from a document. In the following example, the customer number follows a label with the text **Customer Number**. We can configure Ephesoft to extract any numeric text following **Customer Number** into the **Customer Number** field. This process is known as **key/value extraction**. The key is the label, and the value is the text to be extracted.

Staffmark

Bill To:
EFE Company, LLC
14252 Culver Dr. Suite A 145
Irvine CA 92604

Customer Number: 116785
Department :
Invoice No: 0001100953
Invoice Date: 04/30/2009
Amount Due: 446.25 USD
Page: 1

For billing questions, please call your local branch

W/E	Employee / Description	Hours/Units	Rate	Net Amount
04-26-2009	Aguirre, Erika Packaging Clerk	Regular 7.00	12.75	89.25
04-26-2009	Cervantes, Michelle Clerk	Regular 7.00	12.75	89.25
04-26-2009	Chavez Zamudio, Rober Clerk	Regular 7.00	12.75	89.25
04-26-2009	MORENO, GLORIA Clerk	Regular 7.00	12.75	89.25
04-26-2009	Ponce, Sofia Packaging Clerk	Regular 7.00	12.75	89.25
Regular Hours		35.00		
Over Time Hours		0.00		
Other Hours		0.00		
Total Hours		35.00		
Regular Amount		446.25		
Over Time Amount		0.00		
Other Amount		0.00		
Total Amount		446.25		
SUBTOTAL:				446.25
TOTAL AMOUNT DUE :				446.25

The sample invoice form

To define a key/value extraction rule, navigate to the **KV Extraction Rule** area of the menu on the left side of the batch class administration screen. Click on the **Add** button to open the key/value rule screen for a new rule.

In this screen, indicate on an actual invoice where the key and the value are located. Do this by dragging the invoice into the **KV Test** area at the bottom of the screen. The first page of the invoice will fill the right side of the screen, with two colored rectangles. Drag the green rectangle so that it surrounds the label, as shown in the following screenshot. Then, drag the red rectangle so that it surrounds the value.

The screenshot shows the Ephesoft software interface. On the left is a configuration panel for a rule named 'BC4 :: Invoice :: CustomerNumber'. It includes fields for 'Key' (set to 'Customer Number'), 'Value' (set to '{d(6)}'), 'Fuzzy %' (10%), 'Fetch' (FIRST), 'Page' (FIRST), 'Zone' (TOP), and 'Weight' (1). The main area displays an invoice from Staffmark. The invoice header includes the Staffmark logo and the word 'INVOICE'. Below this, the 'Customer Number' is highlighted with a green rectangle and the value '116785' is highlighted with a red rectangle. The invoice also shows 'Bill To' information for EFE Company, LLC. A table of employee data follows, with columns for W/E, Employee / Description, Regular, Hours/Units, Rate, and Net Amount. The table lists five employees: Erica Aguirre, Michelle Cervantes, Rober Chavez Zamudio, GLORIA MORENO, and Sofia Ponce. At the bottom, a 'SUBTOTAL' line shows a net amount of 446.25.

Field extraction configuration

[

The key area should be the approximate size of the key size.
The value area should be as large as necessary to include the largest value that might appear on the page. However, if you make this area too large, you run the risk of Ephesoft including other text from the page in the field value. You can mitigate this risk by using more specific regular expressions for the value.

]

The following information can be entered for each rule:

- **Use Existing Field For Key:** This checkbox indicates that the value from another key/value rule will be used as the key for this rule. This is discussed in the next chapter.
- **Key Pattern:** A regular expression that can be used to find the key. The pattern can be keyed in, selected from the regular expression pool (pre-existing patterns), or built using a regular expression builder.

- **Value Pattern:** A regular expression that defines the format of the content to extract. This can be specified in the same way as the key pattern.
- **Fuzzy %:** This is the amount of text that can be different from the specified key pattern. For example, if an extraction rule has a key pattern of **Customer Number** and Ephesoft discovered **Cust0mer Number** on the document (most likely due to an OCR error), the rule would still fire if the fuzzy percentage level were set to 10%. If the fuzzy % has a selection other than none, the key matching will also be case insensitive.
- **Fetch:** Sometimes, there are several instances of a value that follow a key. Perhaps, the form has a label that says **SSN List** and the text following that is **123-45-6789 987-65-4321**. If the value regular expression is correctly set to match a social security number and the fetch value is **FIRST**, then Ephesoft will extract the value **123-45-6789**. If the fetch value is **LAST**, Ephesoft will extract **987-65-4321**. If the fetch value is **ALL**, then the extracted value will be **123-45-6789 987-65-4321**.
- **Page:** Ephesoft can apply this rule to all the pages in a document or to just the first or last page.
- **Zone:** Ephesoft can apply this rule to the content found in a particular area of the page, such as the top or the side.
- **Weight:** There can be multiple KV extraction rules for a single index field. The most confident extracted value will be used as the index value. The weight allows you to reduce the confidence of some rules. The value for weight must be a decimal between 0 and 1.

In the preceding example, we extract the customer number by using the key pattern of **Customer Number** and the value pattern of `\d{6}`. This regular expression indicates that we expect to see a number that is exactly six digits long.



Patterns are defined using Java regular expressions, the reference for which is found on Oracle's website.

When using longer key values, it is a good idea to set **Fuzzy %** to at least 10%. This will ensure that if the OCR is not perfect, it will still trigger the KV extraction rules.

Creating a Batch Class

Click on the **Test KV** button. The results for the extraction are displayed and the values highlighted in the screenshot:

Ephesoft Apply Test KV Clear Cancel View OCR Data Image Invoice 1.tif Page No. 1

BC4 :: Invoice :: CustomerNumber

Use Existing Field For
Key: ☐
Key: *
Customer Number
Value: *
10(6)
Fuzzy %:
10%
Fetch: FIRST
Page: FIRST
Zone: TOP
Weight: *
1

Bill To:
EFE Company, LLC
14252 Culver Dr. Suite A 145
Irvine CA 92604

Customer Number: 116785
Department: 0001100953
Invoice No: 04/30/2009
Invoice Date: 446.25 USD
Amount Due: 1
Page:

For billing questions, please call your local branch

W/E	Employee / Description	Hours/Units	Rate	Net Amount
04-26-2009	Aguirre, Erica Packaging Clerk	Regular 7.00	12.75	89.25
04-26-2009	Cervantes, Michelle Clerk	Regular 7.00	12.75	89.25
04-26-2009	Chavez Zamudio, Rober Clerk	Regular 7.00	12.75	89.25
04-26-2009	MORENO, GLORIA Clerk	Regular 7.00	12.75	89.25
04-26-2009	Ponce, Sofia Packaging Clerk	Regular 7.00	12.75	89.25

Regular Hours Over Time Hours Other Hours Total Hours

Advance KV Test

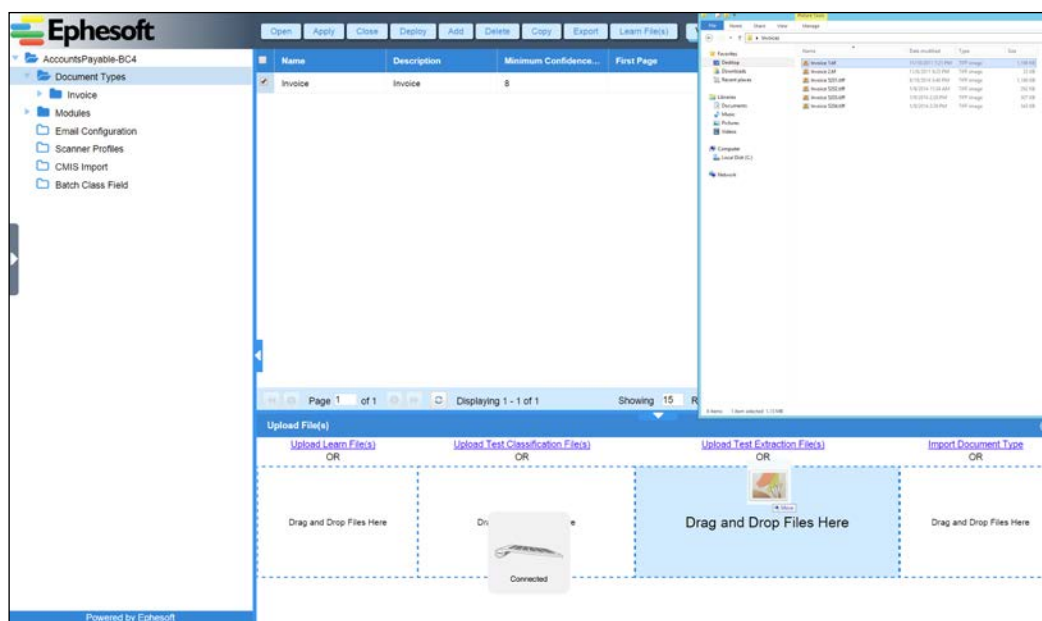
Key	Value	CONFIDENCE%	KEY-COORDINATES	VALUE-COORDINATES	COLORCODE
Customer Number	116785	100	(1395,274)(1725,303)	(1871,273)(1988,304)	

Powered by Ephesoft

The extraction test results

The patterns may need to be adjusted in order for the rule to work properly. Once the rule is working as desired, click on the **Clear** button and then, the **Apply KV**. Click on **Apply** again to save the configuration to the server.

This only tests one rule at a time. Once you get all your document fields and rules set up, you can test the classification of your document and all the rules together by returning to the **Document Types** area of the hierarchy on the right side of your screen. Drag the sample invoice to the area below the **Upload Test Extraction File(s)** link.

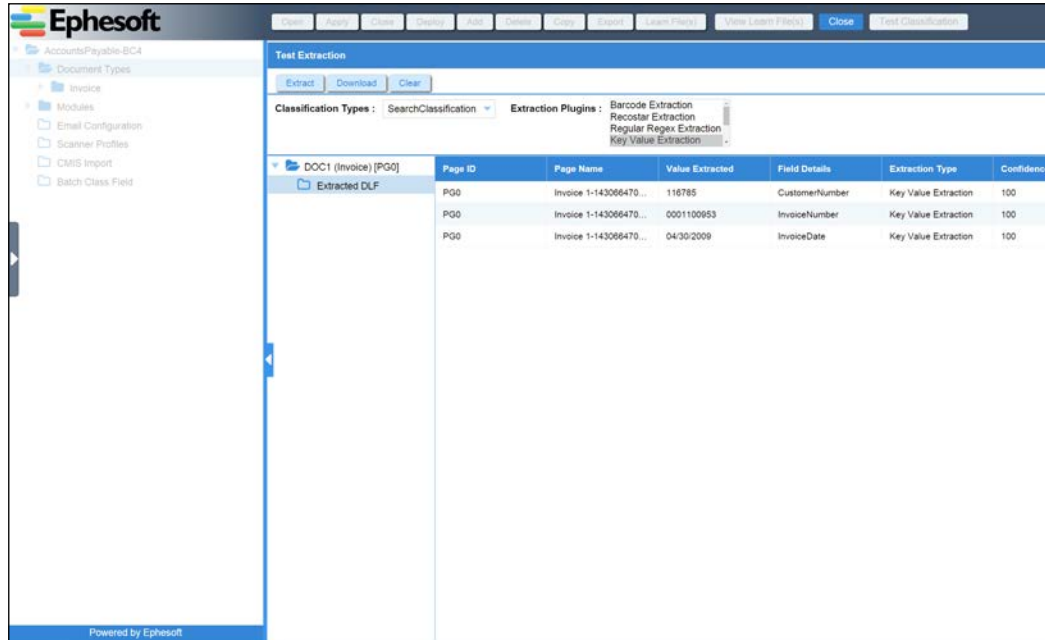


Drag and drop extraction testing samples

After uploading the sample, click on the **Test Extraction** button. Ephesoft will expect you to specify a classification type and indicate which types of extraction to attempt. The default classification type is **SearchClassification**; this is correct for this example. We will discuss other types of classification in the next chapter. We are attempting to test the key/value rules that we set up, so we will select **Key Value Extraction** and then, click on the **Extract** button.

Creating a Batch Class

The results will be displayed for each sample document uploaded.



The screenshot shows the Ephesoft Test Extraction user interface. On the left is a sidebar with a tree view containing 'AccountsPayable-BC4', 'Document Types', 'Invoice', 'Modules', 'Email Configuration', 'Scanner Profiles', 'CMS Import', and 'Batch Class Field'. The main area is titled 'Test Extraction' and includes buttons for 'Extract', 'Download', and 'Clear'. Below these are dropdowns for 'Classification Types' (set to 'SearchClassification') and 'Extraction Plugins' (listing 'Barcode Extraction', 'Recostar Extraction', 'Regular Regex Extraction', and 'Key Value Extraction'). A table displays the extraction results for a document named 'DOC1 (Invoice) [PG0]' under the 'Extracted DLF' folder. The table has columns for Page ID, Page Name, Value Extracted, Field Details, Extraction Type, and Confidence.

Page ID	Page Name	Value Extracted	Field Details	Extraction Type	Confidence
PG0	Invoice 1-143066470...	116785	CustomerNumber	Key Value Extraction	100
PG0	Invoice 1-143066470...	0001100953	InvoiceNumber	Key Value Extraction	100
PG0	Invoice 1-143066470...	04/30/2009	InvoiceDate	Key Value Extraction	100

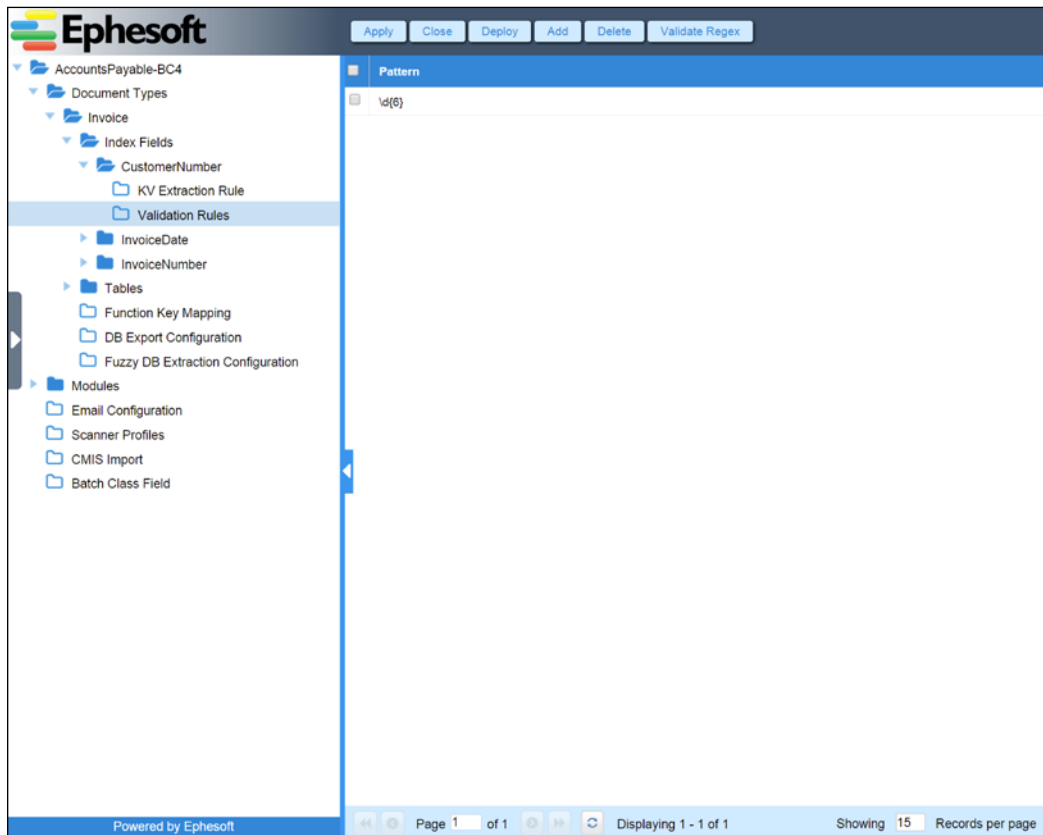
The extraction test results

Once the results are validated, click on the **Close** button to close the **Test Extraction** user interface.

Validation rules

Sometimes, content needs to be checked for consistency before being saved. For example, it may be important to save all dates in the same format, even though they are provided in a variety of formats in the documents. Ephesoft accomplishes this with validation patterns that can be set on a field level. If the field's extracted value does not match the pattern, Ephesoft will force the operator to edit the field to conform. This ensures that values that have a pattern, such as dates and order numbers, are consistent prior to being exported.

A validation rule is simply a regular expression that the field must match. To create a validation rule, navigate to **Validation Rules** on the left-hand side of the batch class administration screen and click on the **Add** button. Enter a regular expression to which the field must conform, and then click on the **Validate Regex** button. In the following example, the administrator is enforcing that the customer number must be a number with exactly six digits. Once validated, click on the **Apply** button to save your changes.



The field validation rules

Export

Most organizations need to send their documents from Ephesoft to some other system. This is the final step in the mailroom workflow that we used as a template.

Ephesoft comes with several plugins available for exporting purposes. Some plugins are of general use, such as the DB Export plugin or the CSV File Creation plugin, while others are targeted at specific ECM products, such as the IBM CM plugin and the FileBound export plugin. There is also a CMIS plugin that allows export to any repository that supports version 1.0 or 1.1 of this industry standard interoperability interface.

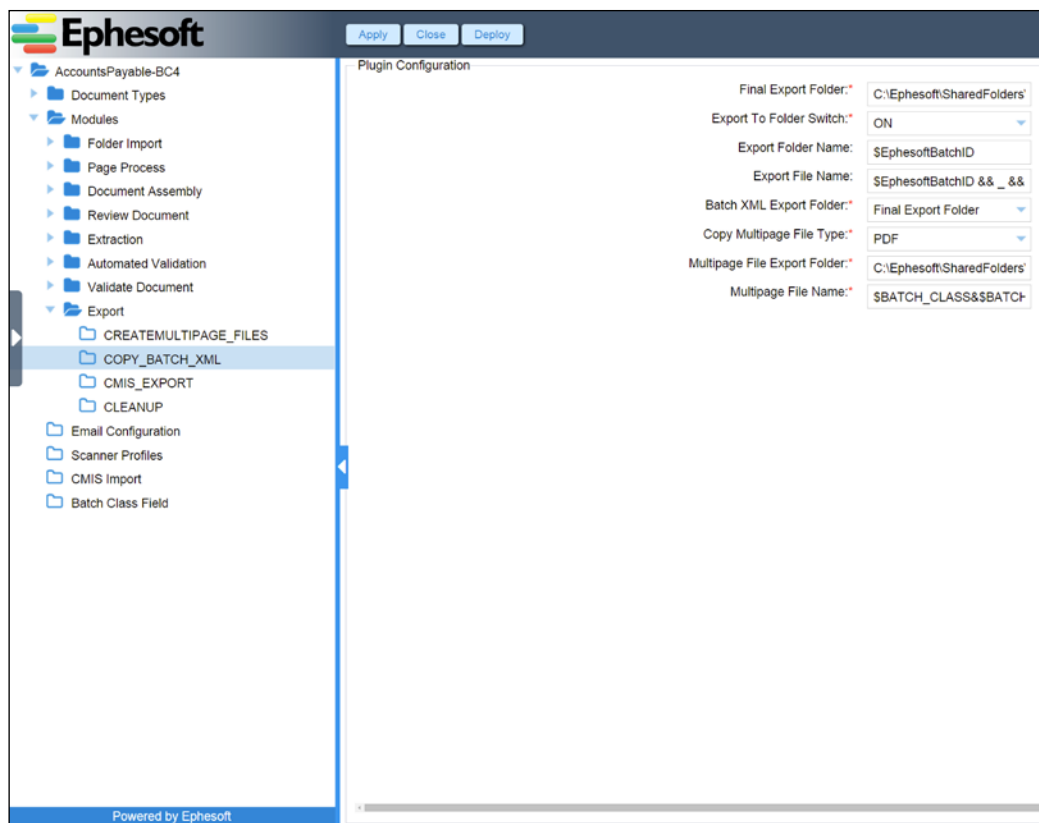
More than one plugin can be active for each batch class, allowing you to export to multiple locations or systems. If no export plugin (or combination of plugins) meets your needs, it is possible to implement your own export plugin.

Copy Batch XML is the only export plugin that is enabled by default.

Copy Batch XML

The Copy Batch XML plugin copies the processed documents to a location on the Ephesoft server's file system. An XML file is placed alongside the documents. This XML file contains the extracted fields and tables for each document, as well as information about the batch.

By default, this plugin is enabled and copies files to `final-drop-folder` within the `SharedFolders` folder of your Ephesoft installation. If you want to disable the plugin, or if you want to change the drop location, you can edit the plugin configuration. From the administrative interface, select and edit your batch class, the export module, and then, the `COPY_BATCH_XML` plugin.



Upload file user interface

The following information can be entered for each rule:

- **Final Export Folder:** This is the base location for all content exported by this plugin.
- **Export to Folder Switch:** This is used to enable or disable this plugin.
- **Batch XML Export Folder:** Select **Batch Instance Folder** to place the batch XML in the folder defined by export folder name. Select **Final Export Folder** to place XML directly under the final export folder
- **Copy Multipage File Type:** This specifies the output format. Options include **PDF**, **TIFF**, or **both PDF and TIFF**.
- **Multipage File Export Folder:** This is the folder destination for the actual documents (not the batch XML).
- **Multipage File Name:** This is the format for the exported document file names.

File and folder names can be assembled using the ampersand (&) operator to concatenate static text with variables. If a piece of the file or folder name begins with a dollar sign, Ephesoft will treat this text as a variable name. The variable can be a document-level field value, or it can be one of the following variables defined by Ephesoft:

- BATCH_CLASS: This is the batch class
- DOCUMENT_IDENTIFIER: This is the document identifier
- BATCH_IDENTIFIER: This is the batch identifier
- DOCUMENT_TYPE: This is the document type
- DATE: This is the date the batch was exported
- TIME: This is the time the batch was exported

The TIME and DATE variables can be modified with expressions that control the format of the text that is rendered, as follows:

`$DATE (yyyy-MM-dd)`

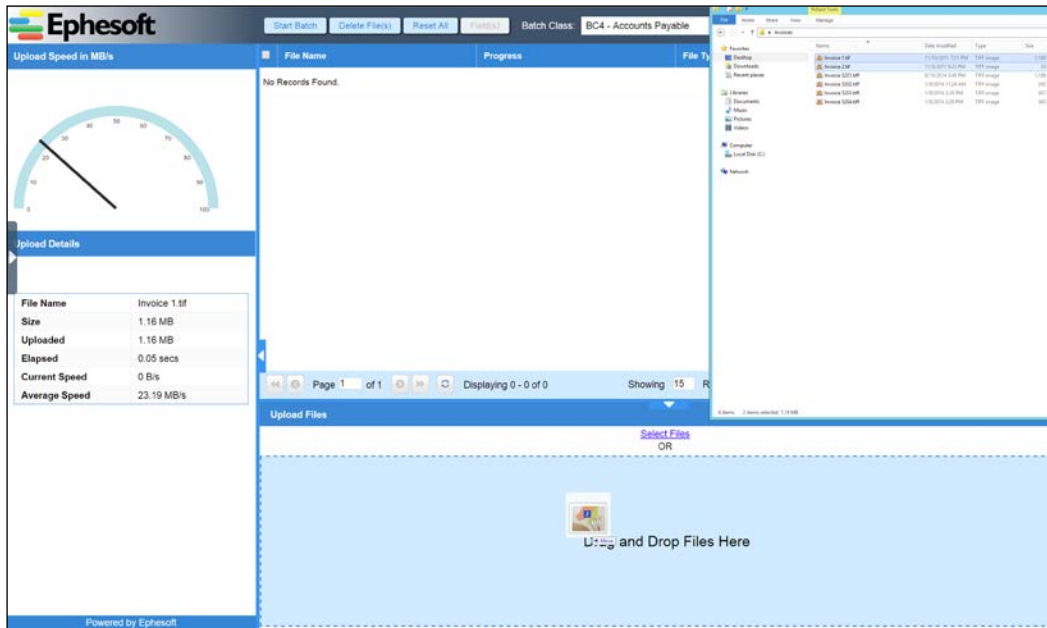
Processing a batch

Now that we have defined our batch class and trained the batch class to recognize documents and extract data from the document, it is time to test it out. To do this, we will turn our attention to the operator's interface to process a batch. To process a batch, we will walk you through the following areas:

- Starting a batch
- Reviewing documents
- Verifying the extracted content

File upload

First, you will need to find some test content. If you already have page images, you can upload the images to Ephesoft. Move your mouse all the way to the left of the screen to reveal the navigation area, and then click on **Upload Batch**. Drag a batch of invoice images to the drop area, select the **Accounts Payable** batch class, and click on the **Start Batch** button.

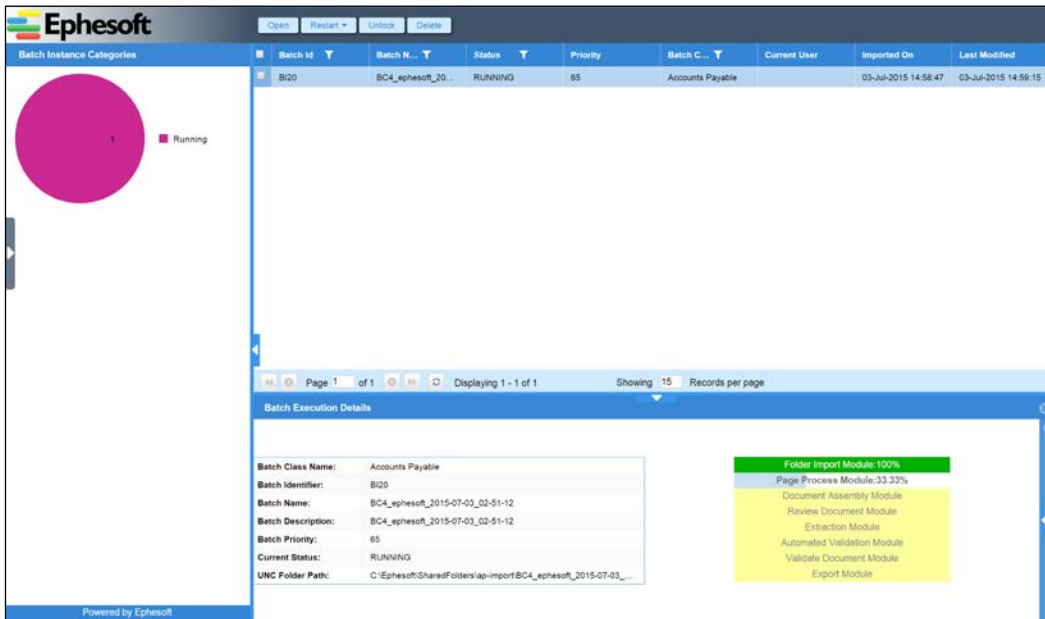


The upload file user interface

Ephesoft should shortly begin to process the batch. You can monitor Ephesoft's progress by using the **Batch Instance Management** tool that is accessible in the navigation menu that is exposed when you move your mouse all the way to the left of the screen.

Creating a Batch Class

It takes a few seconds for the batch instance to be created, so you may not see anything at first. Press the refresh button at the bottom of the list to update the list of batch instances. Once Ephesoft creates the batch instance, you will see an entry with the status of **NEW**. The batch instance will quickly move to **RUNNING** when Ephesoft begins processing the batch.



The batch instance management user interface

If the documents in the batch were scanned clearly and with sufficient resolution, the textual content should be OCR'd with high accuracy. Ephesoft will compare the text on each page in the batch with the text on the pages provided for training. If enough text matches, Ephesoft will classify the document as being of that type.

If Ephesoft's calculated confidence is below the minimum confidence threshold that was configured for the document type, then the operator will have to review the batch. Otherwise, the batch will run all the way through the workflow without any human interaction. Once the workflow is complete, the batch instance will disappear from the list on the administrative **Batch Instance Management** page.

Starting a batch from other sources

Most commonly, content is scanned using a high-volume production scanner or electronic files written to a network-accessible storage device. When you defined your batch class in the previous section, you specified a file system path that should be monitored for new content. In production, this is often a network file path to the location where the scanner writes page images.

If your test content is in a hard copy and you have a TWAIN or PaperStream compatible scanner connected to your computer, you can scan images directly to Ephesoft by using the Web Scanner feature of the operator's interface.

Ephesoft can be configured to periodically check an e-mail account for new e-mails. These e-mails (and their attachments) will be processed as batches. We will show you how to configure Ephesoft to ingest content from an e-mail account later on.

When a new batch is detected, irrespective of its source (e-mail, web scanner, upload, or drop folder), Ephesoft will create a batch instance for the processing of this batch. Each batch instance is assigned an identifier. The batch instance identifier begins with the letters BI followed by a number.

Review and validation

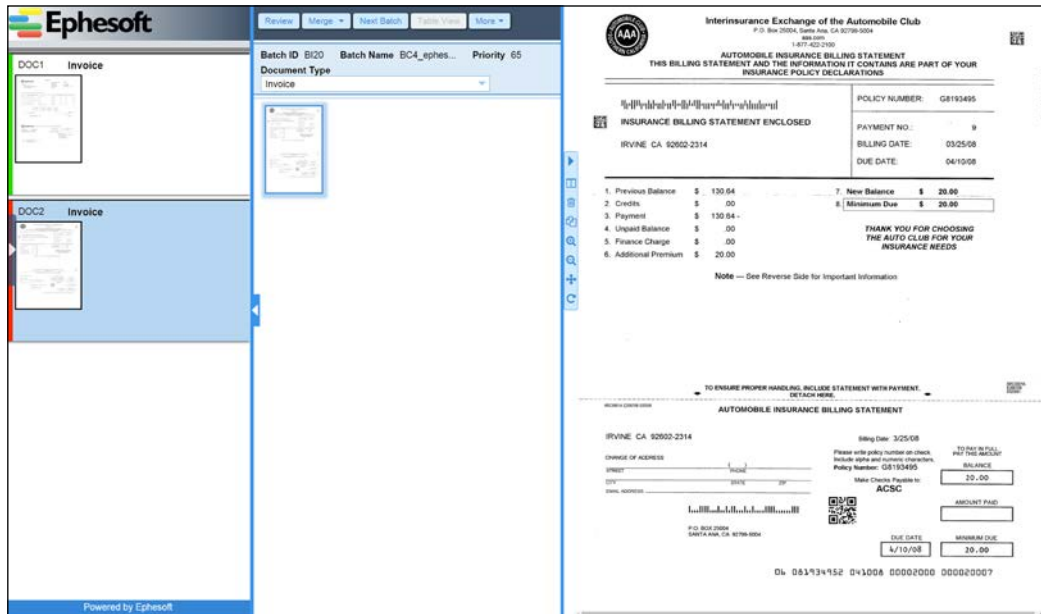
Often, Ephesoft processes each batch without human intervention. Administrators can look at the Batch Instance Management section of the administrative interface to monitor Ephesoft's progress in processing a batch. There are, however, two places where an operator may need to assist Ephesoft.

The first place is in review, where Ephesoft will ask an operator to confirm the document's classification and separation. This review will only be requested if Ephesoft's confidence is below the threshold that you configured for that document type. Ephesoft will skip the review step for a batch if the confidence is greater than the threshold for each document in the batch.

The other place where an operator may need to assist Ephesoft is in validation, which occurs when a validation rule is violated. Earlier in this chapter, we set up a validation rule saying that a customer number must be of six numeric digits. If the content extracted to the customer number field does not match the field's format, Ephesoft will require an operator to edit the field's value. A batch instance will also stop in review if the OCR confidence for any field is below the threshold set on the index field.

Document review

Ephesoft's review feature allows operators to specify the document type, delete, rotate, merge, and split documents, as well as duplicate pages in the batch. Ephesoft directs operators to review a batch when any document in this batch is below its minimum confidence threshold. Documents in the batch that were separated and classified with confidence levels below the threshold configured for a particular document's type will have a red indicator. Documents that meet or exceed their confidence threshold will have a green indicator. If all documents are recognized and the confidence is above the minimum confidence threshold, then the review step is skipped for the batch.



The review user interface

The operator can use keyboard shortcuts to merge a document with the preceding document (*Ctrl + /*), split a document into two documents (*Ctrl + 2*), rotate page (*Ctrl + r* or *R*), and so on. When an operator would like to save the progress of the batch, he/she can save it by pressing *Ctrl + q* or *Ctrl + e*. All shortcuts can be found by clicking on the **More** menu and selecting the **Shortcuts** menu item.



By saving with *Ctrl + q*, the batch will be saved to memory and every 5th save will be written to disk. By saving with *Ctrl + e*, the batch is saved to disk. *Ctrl + q* is less time consuming and you may want to use it particularly for large batches consisting of many pages.

When correcting the document type, you can start to type the name of the desired document type and it will filter the results. This feature is very useful if there are hundreds of document types or many document types with similar names.

Document validation

Validation is the process of verifying extracted content. Ephesoft directs operators to validate a batch if any field of any document could not be extracted or if that field's validation rule failed. When an operator validates a batch, Ephesoft displays the first document with an invalid field value. Fields that violate a validation rule will have a pink background color until the operator enters a value that conforms to the validation rule.

The operator can enter field values with the keyboard or by selecting text on the document image with the mouse. Selecting text on the document image is accomplished by right-clicking to begin the selection and then right-clicking again on the diagonal opposite corner to end the selection.

Documents in the batch that do not have all required values in the appropriate format will have a red indicator, and documents that have all valid values will have a green indicator.

The screenshot displays the Ephesoft validation user interface. On the left, a list of documents (DOC1, DOC2) is shown with their respective status indicators (green for valid, red for invalid). The main area displays the details of the selected document (DOC1, Invoice). The 'Batch ID' is B20, 'Batch Name' is BC4_aphes..., and 'Priority' is 65. The 'Document Type' is set to 'Invoice'. The 'Invoice Header Info' section shows fields for 'Customer Number', 'Invoice Number', and 'Invoice Date'. The right side of the interface displays a sample invoice from the Interinsurance Exchange of the Automobile Club (IAA). The invoice includes a barcode, policy number (08193495), payment number (9), billing date (03/25/08), and due date (04/10/08). It also lists a balance of \$20.00 and provides a QR code for payment. The bottom of the interface shows the 'Powered by Ephesoft' logo.

The validation user interface

Once all of the values are entered, or extracted, for all documents in the batch, the user can save each document's field values by using *Ctrl + q* or *Ctrl + e*. When the user saves the field values for the last document needing validation, he/she will be asked if the batch is completed and can continue on in the workflow.

Summary

You have now learned the basics of how to configure Ephesoft. You can create a new batch class by copying one of the templates, you can train Ephesoft to recognize your document types, extract content into fields, and configure Ephesoft to export your documents and associated metadata. You know how to use the Ephesoft administrative interface to define how batches should be processed and how to use the operator interface (when necessary) to help Ephesoft process a batch instance.

In the next chapter, you will learn to use features that will allow you to more accurately classify documents, extract more content, and export to different destinations.

3

Core Ephesoft Features

You have now seen and used most of the basic features of Ephesoft. We have shown you the administrative interface and used it to create a batch class and to configure classification, separation, and extraction. We have also shown you how to use the operator's interface to process a batch instance, both in review and in validation.

In this chapter, you will learn about the following:

- Different classification types
- New ways to extract content
- Other techniques for exporting your documents and metadata

Classification

In *Chapter 2, Creating a Batch Class*, we showed you how to configure **search classification** to enable Ephesoft to recognize an invoice document. There are several other classification types available; we will explain these alternatives now.

Classification types

You can select the process that Ephesoft will use to classify documents by editing your batch class, editing the Document Assembly module, editing the Document Assembler plugin module within that module, and then selecting a value for DA Classification Type.

Search

Search classification (also sometimes called **Lucene** classification) is the default classification method and is recommended for most content. When configured to perform search classification, Ephesoft compares the text on each input page to the text on training documents to determine its confidence that a document is of a certain type.

Image

Image classification is the best option when classification cannot be made based on content. This occurs on forms that do not have a lot of text, or where the textual content is unpredictable but the physical appearance (such as layout, graphics, and formatting) is consistent. Credit card applications that are red dropout forms (where only the user-entered text is visible to the OCR engine) are candidates for this classification technique.

Barcodes

Barcodes can be used for documents that vary in content and layout, like white mail (unformatted correspondence received in the mail). If a barcode is found on the page with a name that matches an Ephesoft document type, Ephesoft will set the current document's type to that type.

Automatic

The automatic classification type tells Ephesoft to use the scores of every classification plugin that is enabled. This may be necessary when no single classification technique will suffice for your batch class, but configuring multiple classification plugins will have a negative impact on Ephesoft's performance.

One document classification

One document classification is a variant of automatic classification. It assembles all the pages in the batch into a single document.

Confidence

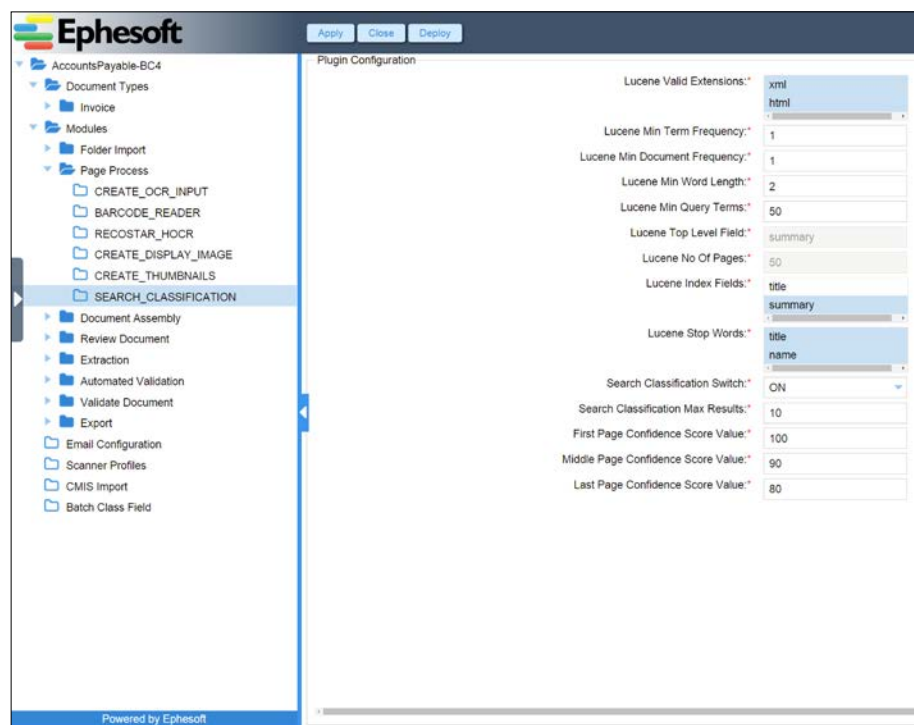
Ephesoft calculates confidence scores for each page in a batch. The page scores represent Ephesoft's certainty that the page being considered is the first, middle, or last page within each document type. They are used to classify and assemble the pages into documents. Ephesoft also uses these page scores to create an aggregate score for each document. This score is compared to the confidence threshold for each document type in the batch class definition. Any document that receives a confidence score below the minimum threshold will be flagged for review. A batch with one or more flagged documents will be placed in a queue for review by an operator.

Confidence scores are calculated differently for each classification type.

Search classification

The default classification type is search classification. Search classification separates and classifies documents by using a two-step process. The first step is to collect information about the pages. The Search Classification plugin of the Page Processing module performs this function. The second step is to separate documents and determine their type. This is the responsibility of the Document Assembler plugin.

- The Search Classification plugin calculates the initial page scores by comparing the text on the page to the text on the training documents. Multiple scores are generated for each page as Ephesoft finds several matches from samples for any given page. The page scores are then adjusted using weighted values that can be modified in the administrative interface by editing the Search Classification plugin of the Page Processing module. Pages can be weighted on the basis of the page type (first, middle, or last). By default, Ephesoft is configured to reduce the scores for the middle and last pages by 10 percent and 20 percent, respectively, as the first pages are more important when it comes to the separation of documents. This effectively biases Ephesoft in favor of using a page to create a new document (over using it as the middle or last page of a document).



The plugin properties of search classification

- Using the page scores calculated in the previous step (and adjusted using the weighted values from the Search Classification plugin), Ephesoft calculates all possible document assemblies and selects the result with the highest score.

The score is calculated as follows: First, the scores of each page in the assembly are averaged. Ephesoft then adjusts the average by using a multiplier in the Document Assembler plugin. You will notice, looking at the following plugin settings screen, that there are several multipliers available. If the assembly has a first and a last page, for example, the *DA Rule first-last Page* multiplier will be chosen. An assembly with the first, last, and middle pages will use the "DA Rule First-middle-last Page" multiplier.

The screenshot shows the 'Ephesoft' application window with the 'Document Assembler' plugin configuration. The left sidebar shows a tree view of the application structure, with 'DOCUMENT_ASSEMBLER' selected. The main area is titled 'Plugin Configuration' and contains various settings for the document assembly process. The settings include confidence thresholds, multipliers for different page rules, and switches for advanced features.

Setting	Value
DA Barcode Confidence	100
DA Rule First-middle-last Page	100
DA Rule First Page	50
DA Rule Middle Page	25
DA Rule Last Page	50
DA Rule First-last Page	75
DA Rule First-middle Page	50
DA Rule Middle-last Page	50
DA Classification Type	SearchClassification
DA Merge Unknown Document Switch	OFF
DA Delete Document First Page Switch	OFF
Advanced DA Switch	ON
DA First Page Confidence Threshold	50
DA Middle Page Confidence Threshold	15
DA Last Page Confidence Threshold	10
Predefined Document Confidence Threshold	0
Predefined Document Type	
Change Unknown Document Type Switch	OFF
Change Unknown Document To Document Type	
Regex Classification Switch	OFF
Regex Classification Pattern	
Regex Classification Default Document Type	

The plugin properties of Document Assembler

Suppose, for example, that you have trained a batch class to recognize the first and middle pages of an invoice. If you run a three-page batch through Ephesoft, you might get the following results:

- Page 1 is determined to be the first page of an invoice because Invoice_First_Page received the highest score:
 - Page 1 compared to Invoice_First_Page receives a score of 30.2
 - Page 1 compared to Invoice_Middle_Page receives a score of 4.2
- Page 2 is determined to be the second page of an invoice because Invoice_Middle_Page received the highest score. Because of the order of this page in the batch, it is determined to be the second page of the invoice found in page 1.
 - Page 2 compared to Invoice_First_Page receives a score of 2.6
 - Page 2 compared to Invoice_Middle_Page receives a score of 12.2
- Page 3 was determined to be the first page of an invoice because Invoice_First_Page received the highest score. Since, it was determined to be a first page, it is the first page of a new document.
 - Page 3 compared to Invoice_First_Page receives a score of 31.6
 - Page 3 compared to Invoice_Middle_Page receives a score of 3.8



In this case, there is no score for Invoice_Last_Page as there were no last page samples used to train this Ephesoft instance.

When using the drag and drop classification training in **Batch Class Management**, Ephesoft will automatically place a last page for any document having more than one page. If that is not the only possible last page of the document type, you will have to go into **Folder Management** and move all samples and files from the last page training for the document type into the middle pages. Once the files are moved, go back into **Batch Class Management** and click on the **Learn Files** button to retrain the system.

The first document assembled will be a two-page invoice because Ephesoft found a first page of an invoice followed by a middle page of an invoice. The second document assembled will be a one-page invoice since only the first page of an invoice was found.

The confidence scores that each of these documents received are calculated as follows:

- Document 1 (page 1 and 2): $(30.2 + 12.2)/2 = 21.2 \times 50\% = 10.6$
 - Average score of pages times the page weight factor, DA Rule First-middle Page
- Document 2 (page 3): $(31.6)/1 = 31.6 \times 50\% = 15.8$
 - Average score of pages times the page weight factor, DA Rule First Page

If the Minimum Confidence Scores setting of the Invoice document type is set to 10, then this batch will skip the review step and move directly to extraction. If the Minimum Confidence Scores for the Invoice document type is set to 15, then this batch will stop in review with the first document requiring review.

Barcode classification

Barcode classification is also a two-step process similar to search classification. In the Page Processing module, pages with barcodes are processed using either the RecoStar plugin or the Barcode Reader plugin. In the Document Assembler plugin, Ephesoft creates documents when the first barcode is found and all the other pages are appended to the document until a new page with a barcode is found. The barcode value found by the barcode or the RecoStar plugin has to match one of the document type names.



On Linux, Ephesoft will always use the Barcode Reader plugin.

Image classification

Image classification compares the pixels on the provided documents to the pixels on the trained documents. The more pixels that match the trained document, the higher is the confidence score that the document will attain. This is in contrast to search classification which OCRs the pages and then compares the text.

When image classification is selected, the Document Assembler plugin uses the image confidence scores to separate and classify documents. The assembly is done using the same algorithm explained in the search classification section.

Automatic classification

Automatic classification uses all enabled classification types. The scores will be combined to come up with an aggregate score per page. This value will be used for assembly and then classification scoring.

Multiple layouts for a single document type

Often, documents of a single type will vary widely in appearance. Invoices from different vendors, for example, will contain similar information, but the format of this information, the layout of the page, and the labels may all vary. When this happens, you must first assess the content of the samples. If the text on the documents is very similar, then it may not be necessary to include samples of each format for classification. Similarly, if the field labels are consistent, it may not be necessary to create new extraction rules for the content.



When analyzing new documents of similar types, you must also decide if it is best to create a new document type (copy an existing document type) or include the document in an existing type. This assessment should take into account how you want to maintain rules and whether the extraction rules would interfere with one another.

Let's consider a case where the content is so different that we must both train the system for classification and separation and create new extraction rules. Compare the following invoice to the one that we have already trained Ephesoft to recognize:

Interinsurance Exchange of the Automobile Club
P.O. Box 25004, Santa Ana, CA 92799-5004
aaa.com
1-877-422-2100

AUTOMOBILE INSURANCE BILLING STATEMENT
THIS BILLING STATEMENT AND THE INFORMATION IT CONTAINS ARE PART OF YOUR
INSURANCE POLICY DECLARATIONS

INSURANCE BILLING STATEMENT ENCLOSED
IRVINE CA 92602-2314

POLICY NUMBER: G8193495
PAYMENT NO.: 9
BILLING DATE: 03/25/08
DUE DATE: 04/10/08

Minimum Due \$ 20.00

1. Previous Balance	\$ 130.64	7. New Balance	\$ 20.00
2. Credits	\$.00		
3. Payment	\$ 130.64 -		
4. Unpaid Balance	\$.00		
5. Finance Charge	\$.00		
6. Additional Premium	\$ 20.00		

**THANK YOU FOR CHOOSING
THE AUTO CLUB FOR YOUR
INSURANCE NEEDS**

Note — See Reverse Side for Important Information

A sample of the second invoice

The new invoice format does not have the invoice number as a 10-digit number to the right of the word **Number**. Therefore, a new extraction rule must be created to obtain this value. In the batch class administration area, edit the Invoice document type, select the **Customer Number** field from **Invoice Fields**, and then, from **KV Extraction**, click on the **Add** button to create a new extraction rule. Enter a key pattern of **POLICY NUMBER** and a value pattern of $G\d{7}$; this is a regular expression for the letter G followed by seven numbers:

Ephesoft BC4 :: Invoice :: CustomerNumber

Use Existing Field For
Key: POLICY NUMBER
Value: G\d{7}
Fuzzy %: 10%
Fetch: FIRST
Page: FIRST
Zone: TOP
Weight: 1

Interinsurance Exchange of the Automobile Club
P.O. Box 25004, Santa Ana, CA 92799-5004
aaa.com
1-877-422-2100

AUTOMOBILE INSURANCE BILLING STATEMENT
THIS BILLING STATEMENT AND THE INFORMATION IT CONTAINS ARE PART OF YOUR INSURANCE POLICY DECLARATIONS

POLICY NUMBER: G8193495

INSURANCE BILLING STATEMENT ENCLOSED
IRVINE CA 92602-2314

PAYMENT NO.: 9
BILLING DATE: 03/25/08
DUE DATE: 04/10/08

1. Previous Balance	\$ 130.64	7. New Balance	\$ 20.00
2. Credits	\$.00	8. Minimum Due	\$ 20.00
3. Payment	\$ 130.64 -		
4. Unpaid Balance	\$.00		
5. Finance Charge	\$.00		
6. Additional Premium	\$ 20.00		

THANK YOU FOR CHOOSING THE AUTO CLUB FOR YOUR INSURANCE NEEDS

Note — See Reverse Side for Important Information

Powered by Ephesoft

Field extraction configuration

Click on the **Test KV** button. You should see your extracted content and the key and value highlighted where it was found. If you do not see your extracted content, you will need to adjust your rules. Once the extraction is performed as desired, click on the **Clear** button, and then, on the **Apply KV** button. Once in the **KV Extraction Rule** table view, remember to click on **Apply** again to save the configuration to the server:

The screenshot displays the Ephesoft software interface. On the left, a sidebar contains configuration options for field extraction, including 'Use Existing Field For Key', 'Key', 'Value', 'Fuzzy %', 'Fetch', 'Page', 'Zone', and 'Weight'. The main area shows an 'INVOICE BILLING STATEMENT ENCLOSED' from IRVINE CA 92602-2314. The invoice includes a policy number G8193495, a payment number 9, a billing date of 03/25/08, and a due date of 04/10/08. A table lists financial items: Previous Balance (\$130.64), Credits (\$0.00), Payment (\$130.64), Unpaid Balance (\$0.00), Finance Charge (\$0.00), Additional Premium (\$20.00), New Balance (\$20.00), and Minimum Due (\$20.00). A note at the bottom says 'THANK YOU FOR CHOOSING THE AUTO CLUB FOR YOUR INSURANCE NEEDS'. At the bottom of the interface, an 'Advance KV Test' table shows the extracted key-value pairs with their confidence levels and coordinates.

Key	Value	CONFIDENCE%	KEY-COORDINATES	VALUE-COORDINATES	COLORCODE
POLICY NUMBER	G8193495	100	(1518,500)(1868,530)	(1962,501)(2152,530)	

The field extraction text results

Once the extraction rules are created and tested for the new invoice format, another test batch that contains a mix of different invoice formats should be run through Ephesoft to verify that separation and extraction are working as desired.

Fuzzy DB

When the fuzzy database is configured, Ephesoft can populate document fields with content from a row in an external database. Ephesoft automatically selects the row whose values match the most content in the current document. Ephesoft uses the Lucene full-text search engine to implement this feature.

Let's configure Ephesoft to populate fields on our invoice documents by using information from a database. Let's assume that we have a database that contains vendor information, including the vendor's name and ID. This vendor ID differs from the customer number that we extracted from the document. First, create the new fields of `VendorID` and `VendorName`.



The database must contain a unique integer value that will be mapped to a RowId value in the Ephesoft configuration (discussed later in this section).

Here, is an example of a database table containing vendor information:

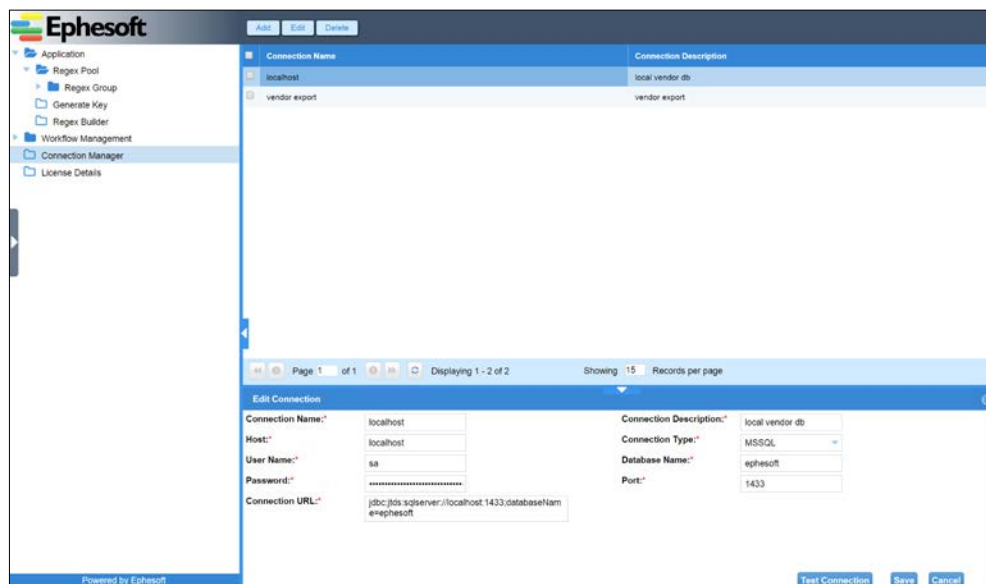
	BDID	Vendor_ID	VendorName	Address	State	City	Zip	Phone	email	webaddress
1	3	33333	ACME	Beverly Hills blvd	CA	Irvine	90210	949-331-7500	NULL	NULL
2	7	77777	ACSC	NULL	NULL	NULL	NULL	NULL	NULL	NULL
3	9	99999	COX COMMUNICATIONS	P.O. BOX 79173	AZ	PHOENIX	85062	NULL	NULL	NULL
4	10	11111	Staffmark	P.O. Box 952396	MO	St. Louis	63195	NULL	NULL	NULL
5	11	22222	Acme Suppliers	123 Sesame Rd	NM	Albuquerque	87105	505-123-4567	NULL	NULL

An example of the vendor database table

We need to configure the fuzzy database plugin to connect to this database. Navigate to **System Configuration** by using the navigation tree that appears when you move your cursor to the far left of the web page. Select **Connection Manager** and click on the **Add** button.



You will need to navigate back to this configuration area after you add the connection. You may want to open System Configuration in a new browser tab. This will enable you to retain the context of your batch class edits in the original tab.



Connection Manager with connection properties for fuzzy database

Enter the required data, and then, click on the **Test Connection** button. When Ephesoft successfully connects to the database, click on the **Save** button.

Navigate back to the batch class management area, select the Invoice document type, and choose **Fuzzy DB Extraction Configuration**. Select **Connection**, previously created in **Connection Manager**, **Table Name**, and **Row ID**:

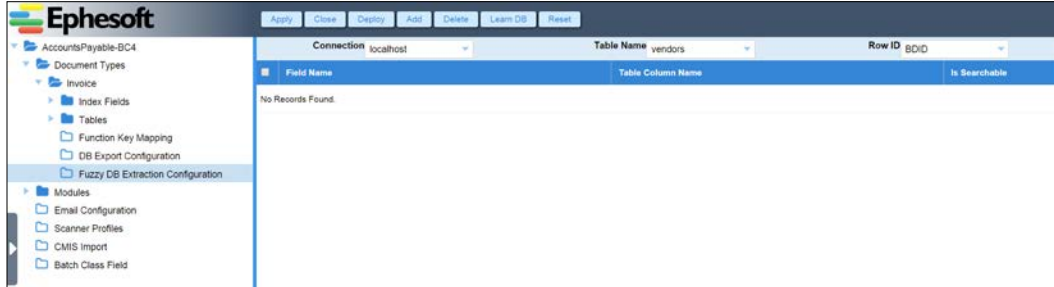




Table configuration of fuzzy database

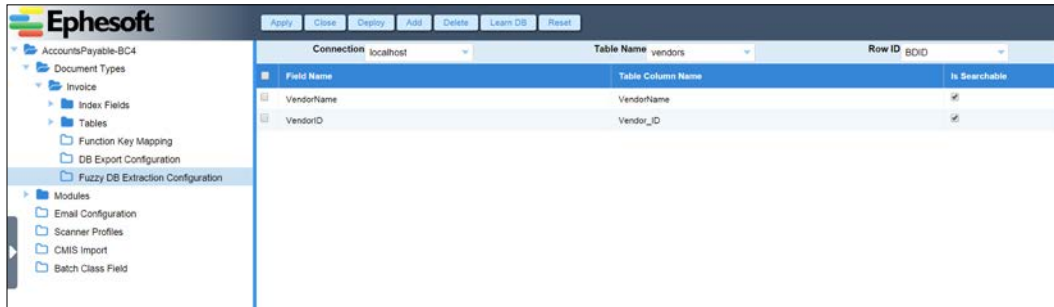
Next, the Ephesoft fields need to be associated with database columns. Click on the **Add** button, and then, set the **Field Name**, **Table Column Name**, and **Is Searchable** fields for each column that you would like to map for the fuzzy database lookup.

 The fuzzy database feature requires that you provide a mapping for the **Row ID** field. As of version 4.0.3.0, this Row ID must be an integer column. Future releases may allow for other unique keys.

Once the fields are mapped to the tables, click on the **Learn DB** button. Click on the **Apply** button to save the configuration.

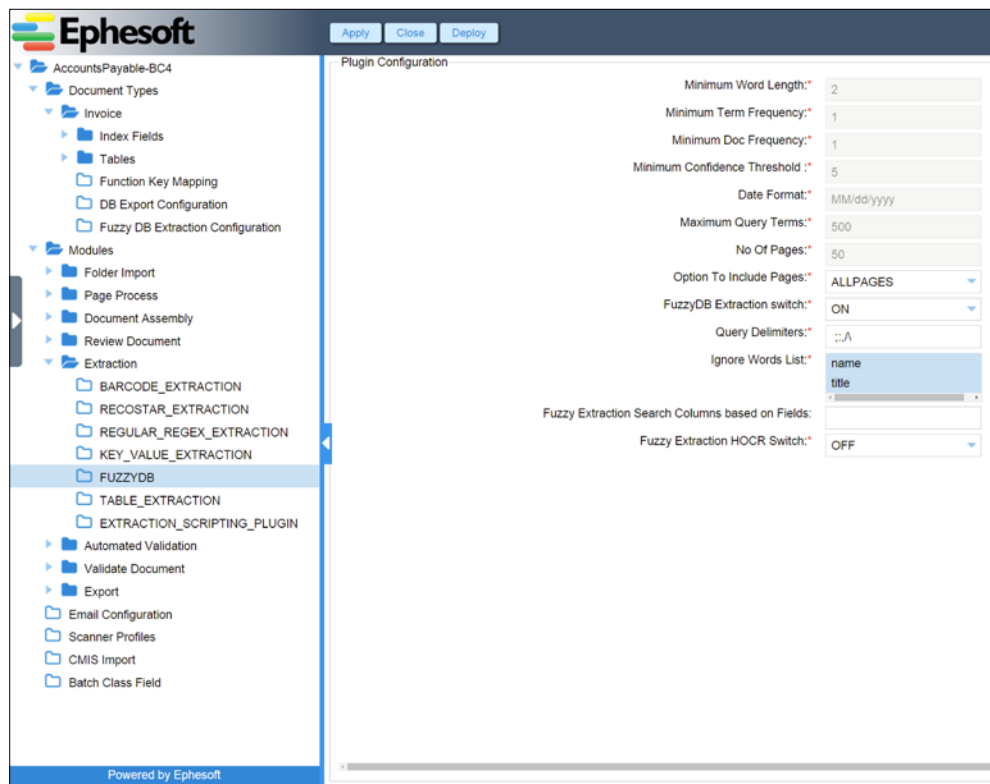
 The **Is Searchable** flag indicates to the system to create the index by using the field(s). This is useful when you want to make a column available for mapping metadata into Ephesoft fields but do not want that field to be part of the search.

After you have added the **VendorID** and **VendorName** fields, the configuration will appear as follows:



Field mapping of fuzzy database

Next, we will turn on fuzzy database extraction. Navigate to the **FUZZYDB** plugin in the **Extraction** module and set the **FuzzyDB Extraction switch** field to **ON**. Click on the **Apply** button:



The plugin properties of fuzzy database

Ephesoft does not operate directly against the database. It uses a Lucene index that is generated from the database. Before using the fuzzy database feature, Ephesoft needs to be instructed to create this index by clicking on the **Learn DB** button.



When your database content changes, you will need to update Ephesoft's Lucene indexes to reflect these changes. You can do this from the administrative interface by clicking on the **Learn DB** button again or you can set up automatic periodic updates by editing Ephesoft's fuzzy-db . properties configuration file.

To test the fuzzy database, create a new batch that contains an invoice with the information in the database. We are using one from Staffmark. On the following screenshot, notice that Ephesoft has determined that Vendor ID should be set to 11111. This is because the text on this page best matched the text in row 4 of the database; both had the word *Staffmark*. Ephesoft uses the values from this row to populate the mapped fields.

The screenshot displays the Ephesoft software interface. On the left, a sidebar shows a list of documents, including two invoices labeled 'DOC1 Invoice' and 'DOC2 Invoice'. The main window is divided into two panes. The left pane shows the 'Batch ID: B125', 'Batch Name: BC4_ephes...', and 'Priority: 65'. Below this, the 'Document Type' is set to 'Fuzzy Search'. The 'Invoice Header Info' section contains fields for 'Customer Number' (116785), 'Invoice Number' (0001100953), 'Invoice Date' (04/30/2009), 'Vendor Name' (Staffmark), and 'Vendor ID' (11111). The right pane displays a detailed view of a Staffmark invoice. It includes the Staffmark logo, invoice details (Customer Number: 116785, Invoice No: 0001100953, Invoice Date: 04/30/2009, Amount Due: 446.25 USD), and a table of billing items. The table lists employees and their respective hours and rates. The total amount due is 446.25 USD.

DATE	Employee / Description	Hourly Rate	Rate	Net Amount		
04-30-2009	Aguirre, Brian	Regular	7.00	12.75	89.25	
04-30-2009	Cervantes, Michelle	Clerk	Regular	7.00	12.75	89.25
04-30-2009	Charles, Robert	Clerk	Regular	7.00	12.75	89.25
04-30-2009	McBride, Gloria	Clerk	Regular	7.00	12.75	89.25
04-30-2009	Ponce, Sofia	Packaging Clerk	Regular	7.00	12.75	89.25

Regular Hours: 30.00 Over Time Hours: 0.00 Other Hours: 0.00 Total Hours: 30.00
Regular Amount: 210.00 Over Time Amount: 0.00 Other Amount: 0.00 Total Amount: 210.00
SUBTOTAL: 446.25
TOTAL AMOUNT DUE: 446.25

A validation sample with extraction results of fuzzy database

There is also an interactive interface to the fuzzy database feature. The operator can use this during validation by entering the text to search for in the **Fuzzy Search** area on the page.

Ephesoft shows a list of database rows that contain the search term. Each row includes a confidence score; this score indicates how well the search term matched the row. When a row is selected and **OK** is clicked, the mapped fields are populated with the information from the index:

The screenshot displays the Ephesoft application interface. On the left, a 'Fuzzy Search Results' window is open, showing a table with the following data:

RowId	VendorID	VendorNa...	Confidenc...
10	11111	Staffmark	54.5

Below the table are 'OK' and 'Cancel' buttons. The main window shows a 'Staffmark' invoice for 'EPE Company, LLC'. The invoice details include:

- Customer Number:** 116755
- Department:** 000100003
- Invoice No:** 04030009
- Invoice Date:** 04/03/2009
- Amount Due:** 446.25 USD
- Page:** 1

The invoice also contains a table of employee hours and a summary of the total amount due, which is 446.25.

Validation of fuzzy search results

Using Web Scanner

Ephesoft provides a tool called Web Scanner that allows you to scan content from nearly any PC that has a scanner attached. You do not have to install and configure scanning software to use this feature, although the scanner drivers must be installed on the PC from which you are accessing Ephesoft. Only TWAIN and PaperStream drivers are supported by Web Scanner.



Because the images are sent directly to the server, bandwidth should be taken into account when using this feature; do not use Web Scanner to process large batches from a device that has a low-bandwidth Internet connection as a primary ingestion scenario.

Web Scanner will not work from a Mac OS X (Apple) client.

If Web Scanner fails to initialize, verify that the popup blocker on your browser is not blocking the location where you installed Ephesoft.

Web Scanner can also be used in the review and validation user interfaces to scan in images to a batch that is already running.

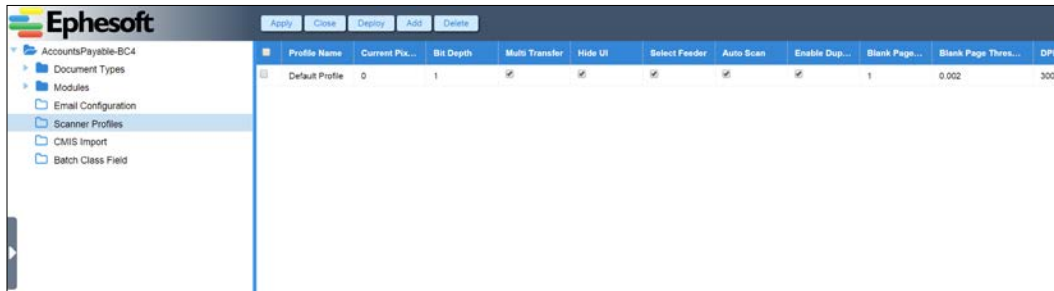
As of version 4.0.3.0, Ephesoft uses a Java Applet to communicate with the scanner. This requires NPAPI to be enabled. A future release will change the Web Scanner to use HTML5 instead of NPAPI.

To set up Web Scanner, click on the **Select Source** button. Ephesoft will display a security dialog. Check the **Always trust content from this publisher** box and click on the **Run** button. Another dialog box will pop up. Select the device from this dialog and click on **OK**. Then, select the batch class workflow to run the scan through and click on **Start**. The scanner will then start, and the operator will be able to see a preview of the scanned images:

The screenshot shows the Ephesoft Web Scanner interface. On the left, there's a sidebar with the Ephesoft logo and a 'Source' button. The main area displays a preview of a scanned document, which is an invoice from Staffmark. The invoice details include: Customer Number: 116785, Invoice Date: 09/03/2015, Invoice Time: 09:03:09, Amount Due: \$448.25 USD. The interface also shows a 'Description' field with 'BC4 - Accounts Payable' and a 'Pages' field with '2'. The bottom of the interface has a 'Powered by Ephesoft' label.

The Web Scanner user interface

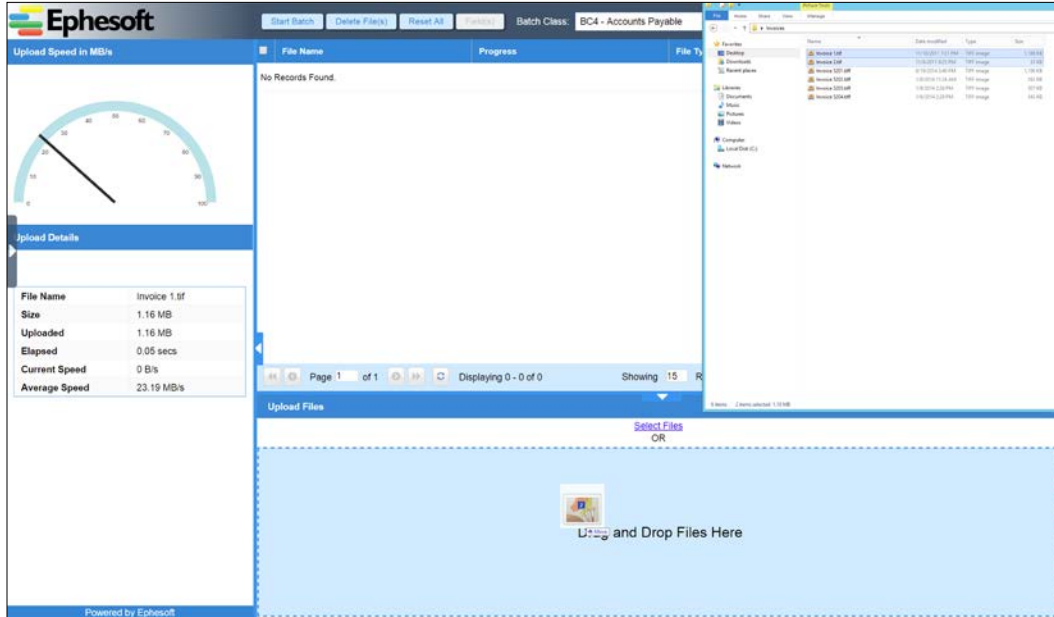
Scanner profiles can be created and edited on a per batch class basis in the Scanner Profiles tab under the Batch Class definition. This allows the user to specify the scanner settings, such as DPI and bit depth, to use when scanning a batch of documents.



The profile properties of Web Scanner


Uploading batches

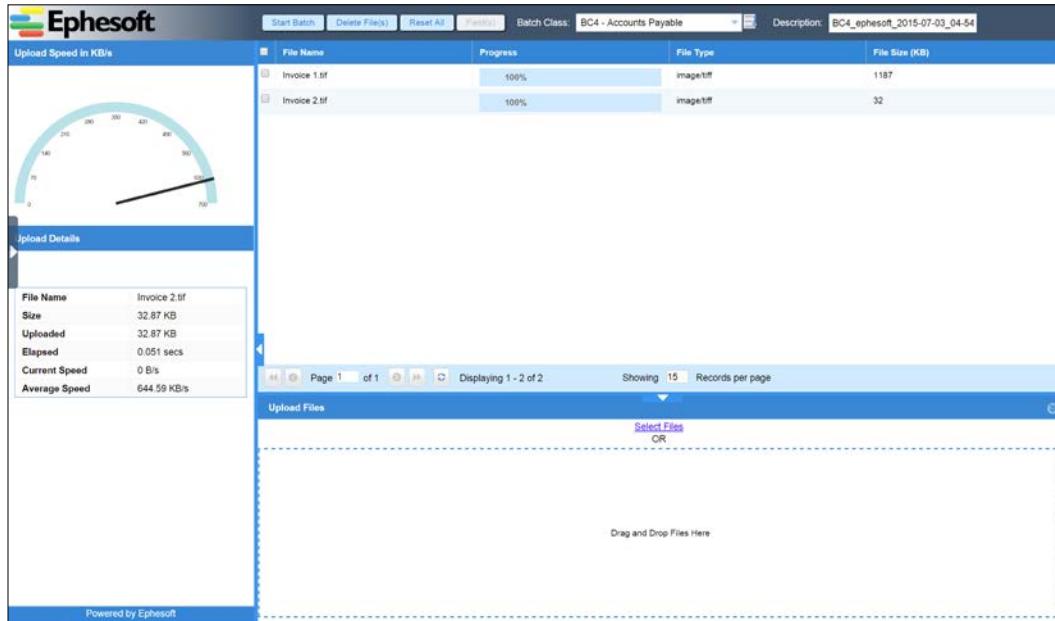
The **Upload Batch** feature allows an operator to select PDF and TIFF images from their local workstation and submit them directly to Ephesoft for processing. Either drag files into the drop area or click on the **Select Files** button to browse the local file system.



The Upload Batch user interface

You can then select the appropriate batch class to process the images and click on the **Start Batch** button to begin processing the batch.

 A description can be provided for the uploaded batch. This information will be displayed in Ephesoft's **Batch Instance Management** area. It is also available in the batch XML.



Upload Batch with uploaded files

Export

In *Chapter 2, Creating a Batch Class*, we used the Copy Batch XML plugin to export content to the Ephesoft server's filesystem. There are a number of additional export options. The CMIS and DB export plugins use standard-based interfaces to allow export to a large number of enterprise content management systems and relational databases. Let's take a look at how to configure these two plugins and then, review the other plugins that are available.

CMIS export

The **Content Management Interoperability Services (CMIS)** API is an open standard for interacting with enterprise document repositories. You can use the CMIS Export plugin to export your scanned content (and associated metadata) to any repository that supports the CMIS standard, such as Alfresco, Documentum, FileNet, or SharePoint. Let's look at how to configure the CMIS Export plugin to send content to Alfresco, a popular open source enterprise content management system.



Ephesoft 4.0 supports CMIS 1.0 and 1.1

Establish a content model in your CMIS

Suppose that you have an Invoice document type in Ephesoft that has fields for Vendor Name, Invoice Date, and Invoice Total. The first thing that you will want to do is define a custom content model in Alfresco to represent your scanned content. Alfresco defines custom content models in XML files that look like the following:

```
<type name="acme:invoice">
  <parent>cm:content</parent>
  <properties>

    <property name="acme:vendorName">
      <title>Vendor Name</title>
      <type>d:text</type>
      <mandatory enforced="false">false</mandatory>
      <index enabled="true">
        <atomic>true</atomic>
        <stored>false</stored>
        <tokenised>false</tokenised>
      </index>
    </property>
```

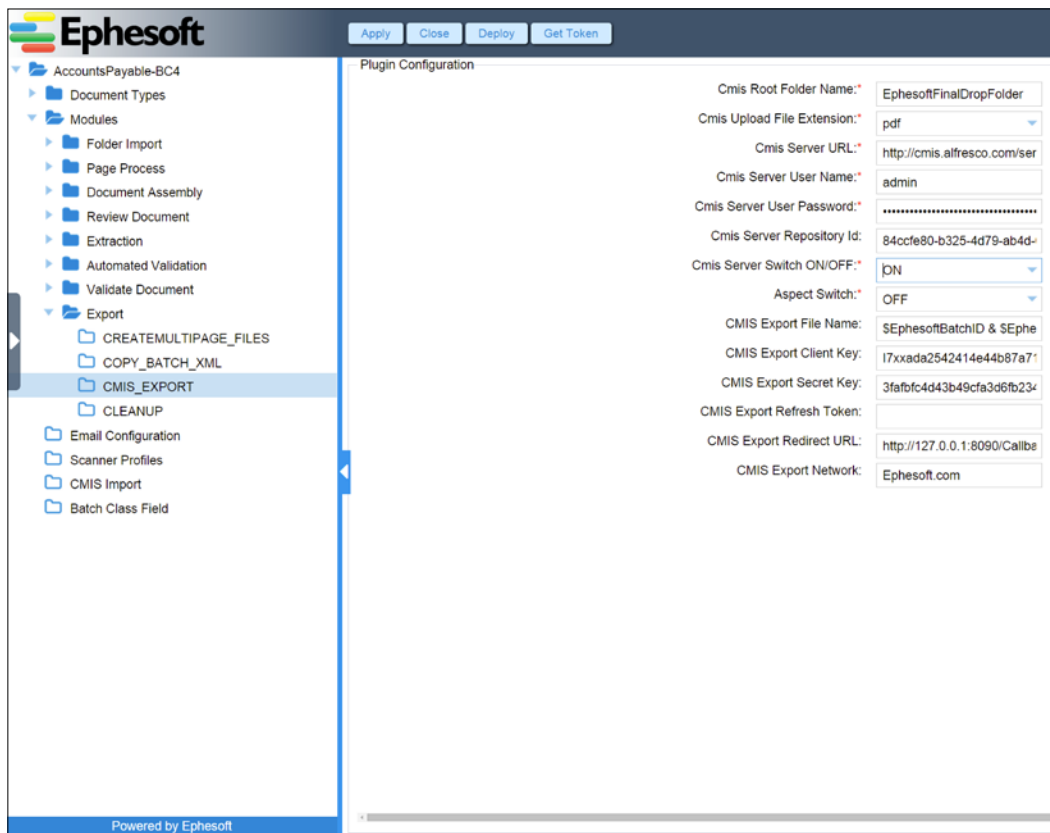
Alfresco document type and property name have prefixes to prevent namespace collisions in the content models. We have used an `acme` prefix in our examples, as would be the case if this implementation were for Acme Corporation. The example above shows a document type `acme:invoice` that extends Alfresco's base document type `cm:content`. This custom type has a text property called `acme:vendorName`. Not shown here are the date property called `acme:invoiceDate` and the float property called `acme:invoiceTotal`.

Configure the CMIS Export plugin

After creating the content model, you will need to configure Ephesoft to use CMIS to send the processed content to Alfresco. There are three places in Ephesoft where you need to configure the CMIS export:

- The plugin settings in the administrative user interface
- The mapping files, in your batch class `cmis-plugin-mapping` folder
- The global configuration file, located in your Ephesoft installation folder here: `Application/WEB-INF/classes/META-INF/dcma-cmis/dcma-cmis.properties`

Let's start with the plugin settings. From the batch class management interface, select and edit your batch class, the export module, and then the CMIS Export plugin. This comes configured by default with a disabled sample connection to Alfresco's public CMIS server.



The plugin properties of CMIS Export

The CMIS plugin can be configured as follows:

- **Root Folder Name:** This is the name of the destination folder in the document repository where Ephesoft should load the exported documents. In Alfresco, this folder will be created underneath the root folder (which is typically named `Company Home`).
- **Upload File Extension:** This setting controls whether the documents are uploaded to your document management system as PDF or TIF images.
- **Server URL:** The services provided by CMIS are defined in an XML service document; this is the location of that document. Alfresco 4.0 hosts this file at `/alfresco/service/cmis`. Alfresco 5.0 hosts this file at `/alfresco/api/-default-/public/cmis/versions/1.1/atom`.
- **User Name and Password:** This is the authentication information required to connect to the document management system.
- **Repository Id:** Some document management systems are capable of hosting multiple repositories. When this is the case, each repository is listed in the service document with an associated identifier. You should examine the service document to find the identifier for your repository.
- **Server Switch:** This can be used to enable and disable export to your document management system.
- **Aspect Switch:** Alfresco manages dynamically assignable groups of properties called `aspects`. This switch enables support for aspects.
- **Export File Name:** Naming convention for the documents exported.
- **Export Client Key, Secret Key, Refresh Token, Redirect URL, and Export Network:** These properties are used to implement OAuth authentication.

Document type and property mapping

Next, you need to associate Ephesoft document types with Alfresco document types. Ephesoft's fields also need to be mapped to the properties of Alfresco documents. Edit this file in your batch class configuration area: `cmis-plugin-mapping/DLF-Attribute-mapping.properties`. This file contains some examples of content mapping. Delete the examples and set up your own mapping, as follows:

```
Invoice=D:acme:invoice
Invoice.VendorName=acme:vendorName
Invoice.InvoiceDate=acme:invoiceDate
Invoice.InvoiceTotal=acme:invoiceTotal
```

The first line of this property file associates the document types, and the last three lines associate the fields. When mapping document types, you will need to prepend `D:` to the beginning of your document repository's type name. This is the CMIS syntax for representing a document (as opposed to, for example, a folder) in Alfresco.

Aspects are configured in the following batch class configuration file: `cmis-plugin-mapping/aspects-mapping.properties`.

Global CMIS configuration

The final area where CMIS is configured in Ephesoft is the following file:

`Application/WEB-INF/classes/META-INF/dcma-cmis/dcma-cmis.properties`.
This file affects the CMIS configuration of all batch classes.

The most commonly modified setting in this file is the date format. When you map a date field, Ephesoft needs to parse the date in order to reformat the information to match the CMIS specification. The `cmis.date_format` parameter specifies how Ephesoft fields that will be exported using CMIS will be formatted. See the JavaDoc for the `SimpleDateFormat` class to learn how to specify date formats.

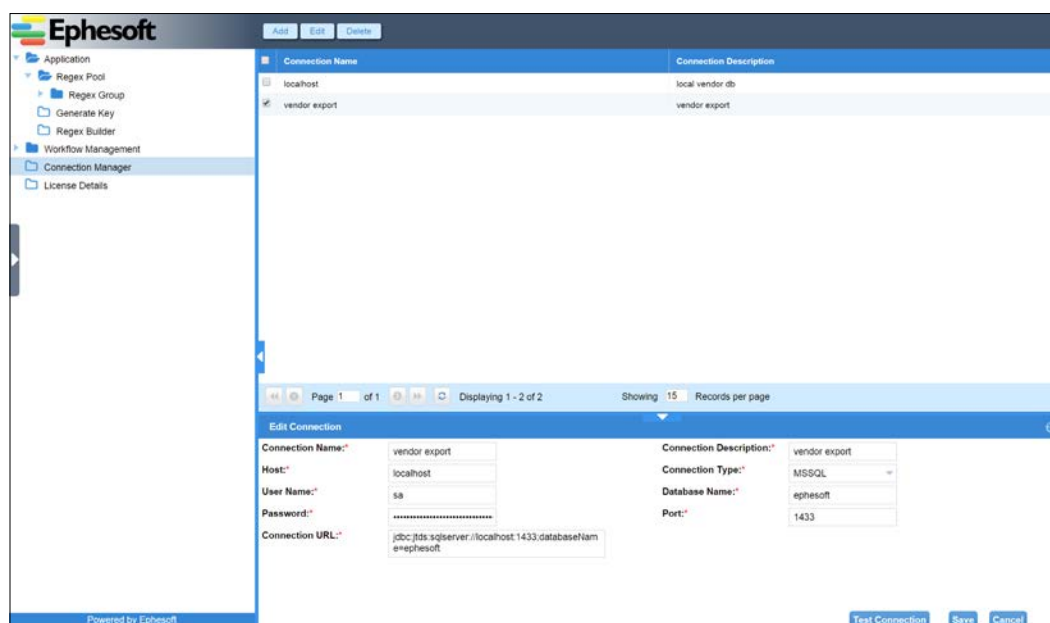
If your content management system uses **Web Service Security (WSS)** to secure its CMIS web services, you will need to adjust the value of the `cmis.security.mode` property. This specifies the security mode to use when attempting to connect to the CMIS web services. There are two possible values: `basic` and `wssecurity`. HTTP Basic Authentication is the default setting for the Ephesoft CMIS connection. This corresponds to the `basic` setting for the `cmis.security.mode` property. The `cmis.security.mode` property is set to `wssecurity` in order to have the CMIS credentials that are configured in the **CMIS_EXPORT** plugin included in the WS-Security SOAP header of the CMIS web service requests.

If your CMIS web services are not addressable from a single URL, you can configure the location of each service used by Ephesoft. You will see a set of properties that begin with `cmis.url`. These can be edited to specify where your content management system hosts this service's WDSL.

Database Export

DB Export allows document level fields values and metadata to be exported to relational databases using JDBC. Administrators can map the Ephesoft document fields to the database table columns.

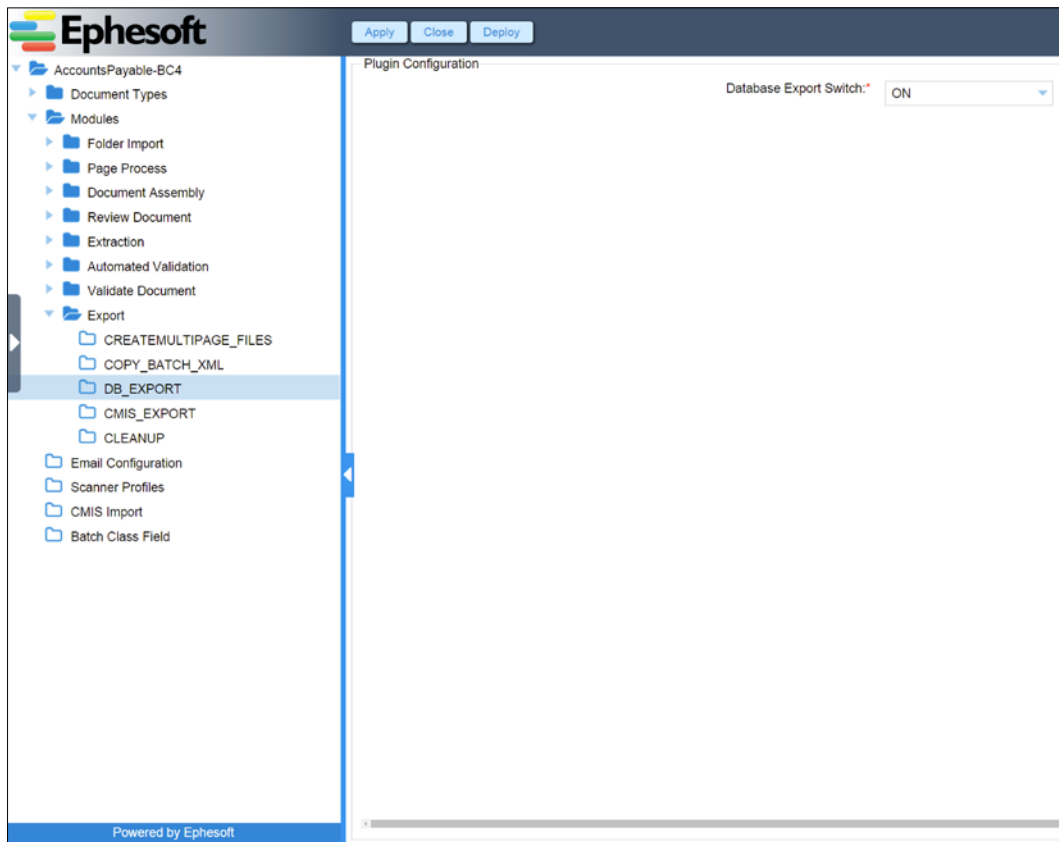
First, go to the system configuration area to create a new connection in **Connection Manager**:



Connection Manager with connection properties for database export

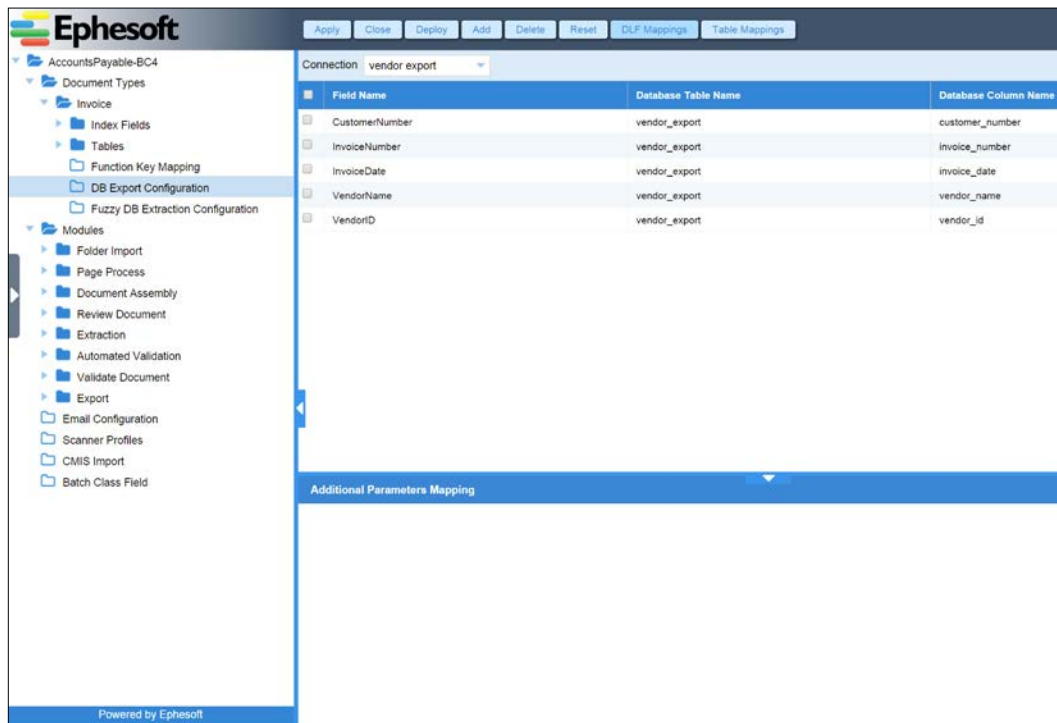
Next, return to the batch class management area and configure your batch class. If the **DB Export** plugin is configured into this batch class' workflow, then you will be able to configure the plugin from the **Modules** section. In *Chapter 4, Ephesoft's Advanced Features*, you will learn how to add plugins to your batch class' workflow.

The configuration of the plugin is simple; there is simply a switch to enable the plugin.



Plugin properties of database export

In **Batch Class Management** under the document type, you can configure **DB Export Configuration**. Select the correct database connection, and then, map the document type fields to the table and column. Click on **Apply** to save your changes.



Database export mapping

When the DB Export plugin runs, it will export the extracted field data for each document in the batch.

Results		Messages			
	customer_number	invoice_number	invoice_date	vendor_id	vendor_name
1	116785	0001100953	2009-04-30 00:00:00.000	11111	Staffmark
2	193495	G8193495	2008-03-25 00:00:00.000	77777	ACSC

Sample results of database export

Other export plugins

Thus far, we have shown you how to export to the local file system or use CMIS and JDBC. These are general-purpose plugins that can be used in a variety of situations. Ephesoft comes with a few other general-purpose plugins such as the CSV plugin and the tabbed PDF plugin.

Ephesoft also provides a handful of plugins to facilitate export into specific content management systems such as DocuShare, HP II FileNet, and IBM CM.

To see the list of available plugins, you should edit your batch class and then edit the export module.

Configuration management of batch classes

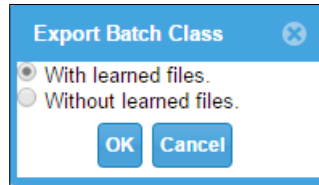
Batch classes can be exported from one Ephesoft server and imported into another. Exporting can also be used to back up the batch class to a code repository such as Git or Subversion. A new batch class can be created or an existing batch class overridden in the importing environment.

To export, select the batch class and click on the **Export** button.

Identifier	Name	Description	UNC Path	Version	Priority	Current User	Encryption Algo	Roles
BC1	MailroomAutom...	Mailroom Auto...	C:\Ephesoft\Sh...	1.0.0.0	1			
BC2	SearchablePDF...	Searchable PD...	C:\Ephesoft\Sh...	1.0.0.0	1			
BC3	GridComputing...	Grid Computing...	C:\Ephesoft\Sh...	1.0.0.0	1			
BC4	AccountsPayable	Accounts Payable	C:\Ephesoft\Sh...	1.0.0.57	65			

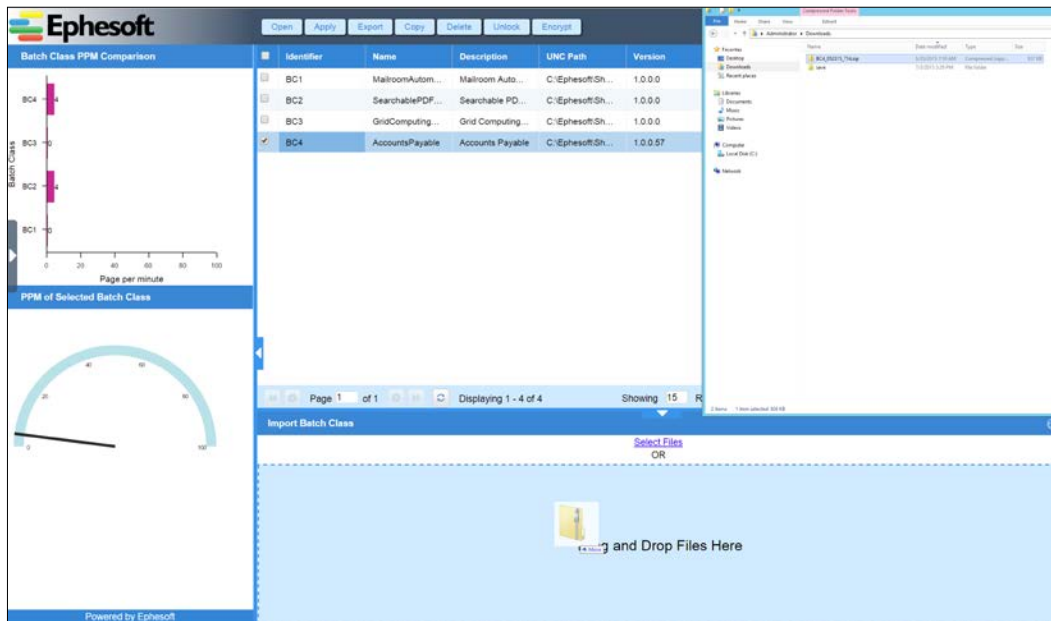
Batch class export

When you click on the **Export** button, Ephesoft will ask you whether you want to export the batch class with or without learned files. The learned files are the images that you used to train Ephesoft to recognize each document type. These files are not necessary for the batch class to operate but can be useful if you want to diagnose errors or re-train Ephesoft. Select **With learned files** and click on the **OK** button. The batch class is saved into a zipped file that is downloaded to your local system.



The batch class export options dialog

Next, import the batch class into an Ephesoft instance. In the **Batch Class Management** screen, drag an exported batch class to the area below the list of batch classes or click on the **Select Files** button to upload the batch class.



Import Batch Class

When you import a batch class, you can create a new batch class by using the imported information, or you can overwrite an existing batch class. To import into a new batch class, deselect the **Use Existing** checkbox, and specify the UNC folder to watch for page images. Click on the **OK** button to create the new batch class.



When exporting the batch class from one environment to another (for example, the test system to the production system), you may not want to include the e-mail accounts and connections; those typically differ for each environment, and in the case of e-mail, could cause issues with two systems monitoring the same e-mail accounts.

The screenshot shows the 'Import Batch Class' dialog box. It has a blue title bar with the text 'Import Batch Class' and a close button. The dialog contains the following fields and options:

- Batch Class Name:** ImportedAccountsPayal
- Batch Class Description:** Imported Accounts Pay
- Priority:** 65
- UNC Folder:** C:\Ephesoft\SharedFolder
- Use Existing:** ☐
- Import Batch Class:**
 - ☒ With Learned Files
 - ☐ Without Learned Files
- CMIS Mappings:**
 - ☒ Export Plugin Properties
 - ☒ Import Configuration
 - ☐ Export Mappings
 - ☐ Roles
 - ☐ Email Accounts
 - ☐ Import Connections
- Encryption:**
 - ☒ Use Existing Key
 - Batch Class Key:** [Empty text box]
 - Algorithm:** None (dropdown menu)

At the bottom are 'OK' and 'Cancel' buttons.

A new Import Batch Class options dialog

Once the batch class is imported, it will appear in the list of batch classes and can be configured as necessary. To replace an existing batch class, upload the exported batch class and select the **Use Existing** checkbox. When you do this, the UNC folder becomes a list of batch classes. Select the batch class that you wish to replace.

Deselect the environment-specific items that should not be overridden. The following settings are often disabled when replacing a batch class:

- **Export Plugin Properties**
- **Import Configuration**
- **Roles**
- **Email Accounts**

Import Batch Class

Batch Class Name: * ImportedAccountsPayal

Batch Class Description: * Imported Accounts Pay

Priority: * 65

UNC Folder: * BC5 - ImportedAccor ☒ Use Existing

Import Batch Class

☒ With Learned Files
☐ Without Learned Files

CMIS Mappings

☐ Export Plugin Properties
☐ Import Configuration
☐ Export Mappings
☐ Roles
☐ Email Accounts
☐ Import Connections

Encryption ☒ Use Existing Key

Batch Class Key:

Algorithm: None

OK Cancel

An existing Import Batch Class options dialog

Click on **OK** to replace the batch class configuration with the settings from the imported batch class.

Summary

In this chapter, you have learned how to process forms with many different layouts, additional extraction techniques, fuzzy database lookup, web scanning, and much more. At this point, you should be able to use Ephesoft to implement intelligent document capture for a wide variety of organizations.

In the next chapter, we will cover more advanced topics such as scripting, web services, and custom workflows.

4

Ephesoft's Advanced Features

Now that you have a strong understanding of Ephesoft, we have saved a handful of advanced topics for last. Advanced topics are those that involve programming skills (as is needed in order to perform scripting and web services) or are only used in special circumstances.

Set aside a little extra time, and get ready to learn the nitty-gritty details of Ephesoft implementation. The following advanced topics will be covered in this chapter:

- Configuring other classification methods
- Additional extraction methods
- Scripting
- Workflow and plugin customizations
- Web services

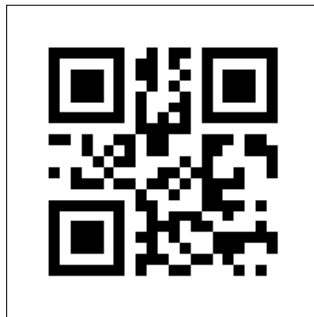
Configuring other classification methods

Search classification was discussed in detail in previous chapters. While search classification is the default and most commonly used classification type, there are other types that can be used instead of or in addition to search classification.

Barcode classification

Barcodes are useful for classifying documents that do not have a consistent format or content, such as correspondence mail. They can also be leveraged for documents that already have barcodes. Setting the classification type to `BarcodeClassification` or `AutomaticClassification` will enable barcodes. The classification type is configured in the Document Assembler plugin of the Document Assembly module. When Ephesoft detects a barcode on a page, Ephesoft converts the barcode to a text value and then looks for a document type with a name that exactly matches this text value. If a match is found, the page will be considered to be a document of this type.

In the invoice example, we can create a QR code that has Invoice as the value:



QR code with the text Invoice

If this code is found anywhere on a page, that page will be classified as the first page of an invoice document.

Ephesoft provides two different plugins that support barcode recognition. These are RecoStar HOCR Plugin and Barcode Reader Plugin. When implementing a Windows-based solution, it is simplest to use RecoStar HOCR Plugin, since this plugin will already be configured in the batch class to enable OCR. For Linux solutions, RecoStar HOCR Plugin is not available, so you must use Barcode Reader Plugin.

Whichever plugin you use, you must enable barcode classification. The plugins are configured in the Page Process module.

- **RecoStar HOCR Plugin:** In order to enable barcode classification using this plugin, you must select `FPR_Barcode.rsp` as **RecoStar Project File Name** and turn **Barcode Switch** to ON.
- **Barcode Reader Plugin:** You may need to add this plugin to your batch class's Page Process module. Once this is done, turn **Barcode Classification Switch** to ON.



Barcodes can also be used for determining the separation of documents. The switch to turn this feature on is in the Document Assembler plugin of the Document Assembly module. Set the value of **DA Merge Unknown Document Switch** to ON.



The supported barcode formats include CODE 29, CODE 93, CODE 128, CODEBAR, DATAMATRIX, EAN13, ITF, PDF 417, and QR.

Image classification

Image classification is another technique for identifying and assembling documents. As with search classification, you will provide samples of the document types. Unlike search classification, when performing image classification, Ephesoft does not attempt to recognize text on pages of the training documents. Instead, the page images are compared to see how many pixels they have in common.

Samples can be dragged into the drop area on the Document Types page or you can use the **Upload Learn File(s)** link to upload samples.

You may need to add the Classify Images plugin to your batch class's Page Process module. Once this is done, turn **Classify Image Switch** to ON. To use image classification, set the classification type to either `ImageClassification` or `AutomaticClassification` in the Document Assembler plugin of the Document Assembly module.

Fixed form extraction

At the beginning of this book, we created a batch class by copying the Mailroom Automation Template batch class. In this batch class, Ephesoft extracted content from the document into fields using key value extraction. Sometimes, it is necessary to use fixed form extraction. This is the process of extracting content from a specific location on the page. Fixed form extraction is configured in the OCR engine, and this configuration is referenced in Ephesoft. Some examples of cases in which fixed form extraction may be used include the following:

- Handwritten content needs to be extracted.
- The form has checkboxes, and the state (whether the checkbox is checked or unchecked) needs to be extracted.

- You need to determine whether there is markup in an area. An example of this is when you want to determine whether a signature line is blank.
- The content to be extracted cannot be located with respect to a key and does not conform to a unique pattern. For example, the form might always have an address in the upper right-hand corner, but there is no label or header associated with this address.



As of version 4.0.3.0, fixed form extraction is implemented using the RecoStar OCR engine. This OCR engine is only used by the Windows version of Ephesoft Enterprise. Therefore, you must be using the Windows version of Ephesoft Enterprise in order to perform fixed form extraction. Ephesoft is working on an implementation of fixed form extraction that uses Nuance's OCR engine; this will make fixed form extraction available on Linux.

You can use fixed form extraction and key value extraction on the same document.



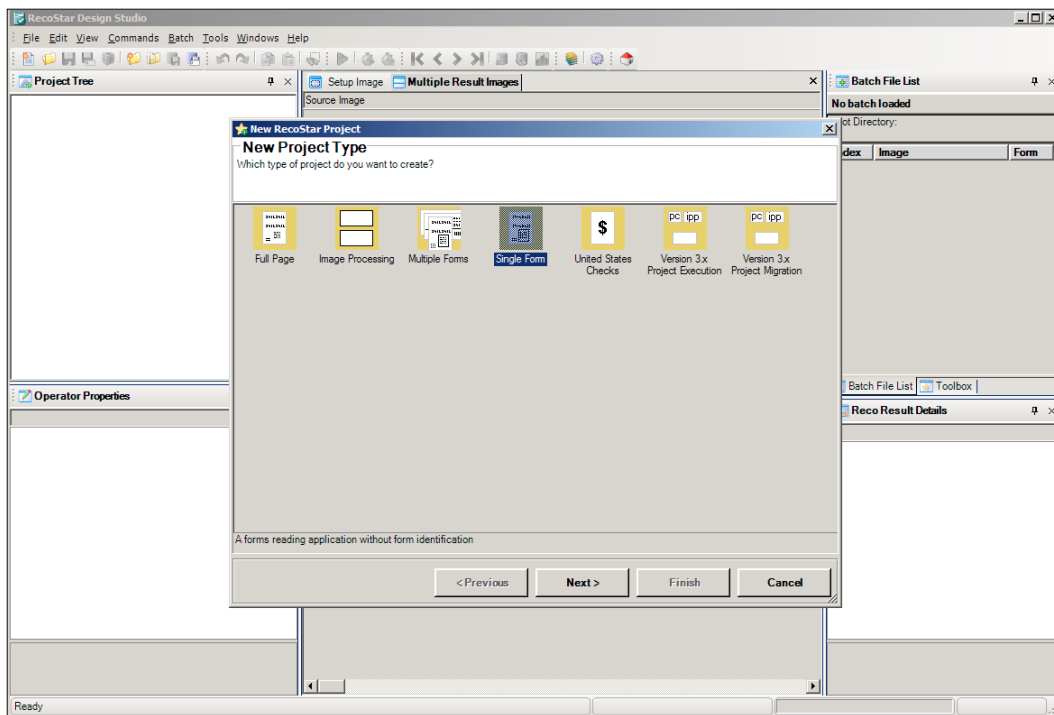
Full Page OCR on Windows is accomplished with a RecoStar project called `Fpr.rsp` located in `SharedFolders\BC99\fixed-form-extraction`. You may want to modify this project if you want to support languages or locales or modify the image manipulation or OCR settings. You can also create a new project to use for the OCR functionality. If you choose to create a new project, you must configure Ephesoft to use that project. This setting is located in the `RECOSTAR_HOCR` plugin in the Page Process module.

Let's walk through the process of using RecoStar to extract a handwritten billing code from an invoice. RecoStar extraction is not configured from within the Ephesoft administrative interface; instead, a Windows application called RecoStar Design Studio must be run. Ephesoft installs this at `Application\native\RecoStarPlugin\RecoStarDesignStudio\RecoStarDesignStudio.exe`.

Creating a RecoStar project

Perform the following steps to create a RecoStar project:

1. Within RecoStar Design Studio, create a new project by selecting **New Project** from the **File** menu.
2. Select the project type **Single Form** and then click on **Next**, as shown in the following screenshot:



New RecoStar project type

3. Enter a project name and location in which to save the file:

New RecoStar Project

New Project File Name

Where shall the new project be installed?

Project Name:

Invoice

Location:

C:\Users\dev\Documents

Browse...

Project Directory (will be created):

C:\Users\dev\Documents\Invoice

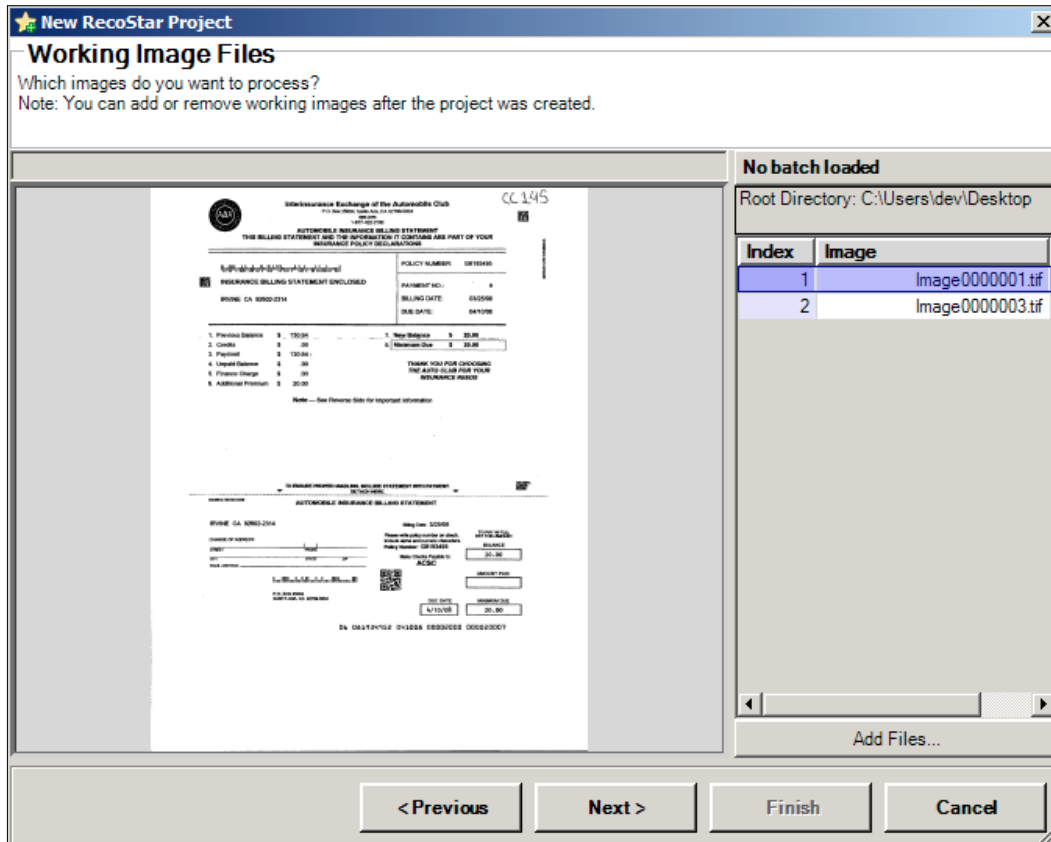
Existing Files/Directories:

- InvoiceFixedForm
- MortgageCheckBox
- My Music
- My Pictures
- My Videos
- Shared Toad
- Smart Touch
- Test1
- Test2
- Test3

< Previous Next > Finish Cancel

RecoStar project properties

4. Use the **Add Files** button to select some working image files. Working image files are sample documents that RecoStar will use to test the configuration. Click on **Next** after uploading the working image files:



RecoStar project sample images

- The next step in the RecoStar wizard for creating new projects is the **Mandatory Parameter** setup. Select the country/languages that appear on the documents to be processed:

New RecoStar Project

Mandatory Parameter

The declaration of Country is mandatory. Please specify.
Note: You can modify this parameter after the project was created.

Selected: 1 FormOperator

Name	Operator
Timeout	180000
SetupImageFileName	
ProjectID	0
DefaultFormType	
ExternalFormType	
Country	USA
FormReading	True
<input checked="" type="checkbox"/> FormGeometry	0 0 0 0 0 0
ResultCoordinates	OriginalImage
ResultImage	Off
ResultGraphicalObjects	False
PassThroughID	Ignore

Country
Selects one or more countries and/or languages.
...

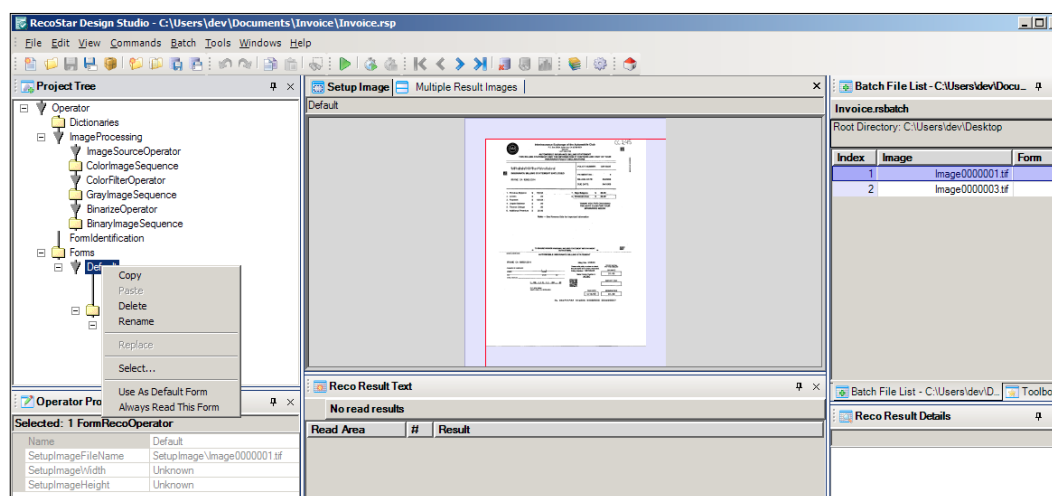
< Previous Next > Finish Cancel

RecoStar project locale selection

- Finally, click on the **Finish** button to create the project.

Configuring the RecoStar project

The RecoStar form name must match the Ephesoft document type name. Find the default form in the project tree and right-click on it. Select **Rename** and change the name from Default to Invoice:



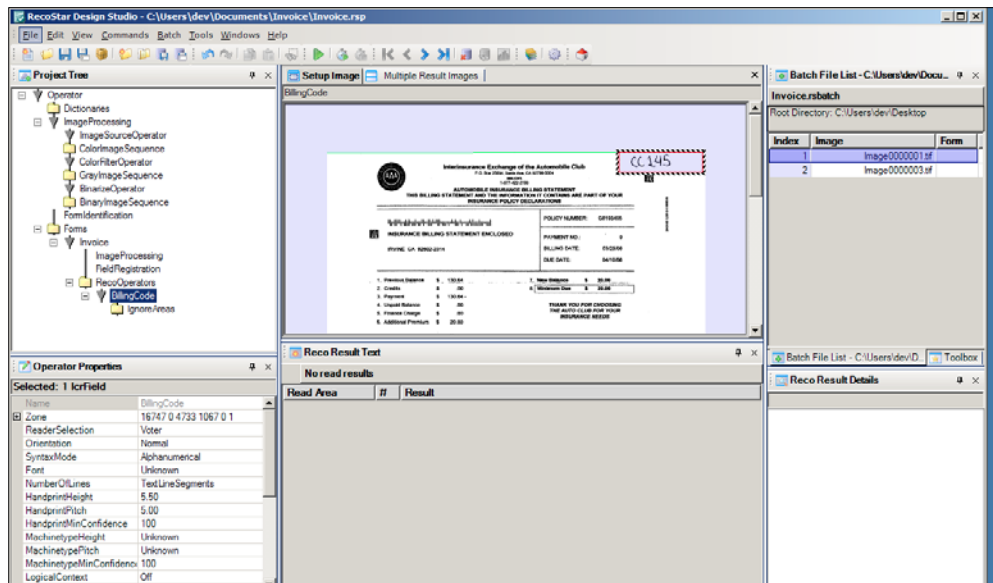
RecoStar form context menu

Next, configure a field to match a field that we will create in Ephesoft. In the project tree, under the default form that we just renamed as *Invoice*, there is a field named *IcrField*. Right-click on this field to rename it as *BillingCode* so that it matches the field name in Ephesoft.

Now, we need to configure RecoStar to show where the billing code is located. Making sure the billing code field is selected in the project tree, select the **Setup Image** tab in Design Studio. Drag the red rectangle in the **Setup Image** tab so that it surrounds the top-right section of the invoice where the billing code is written.

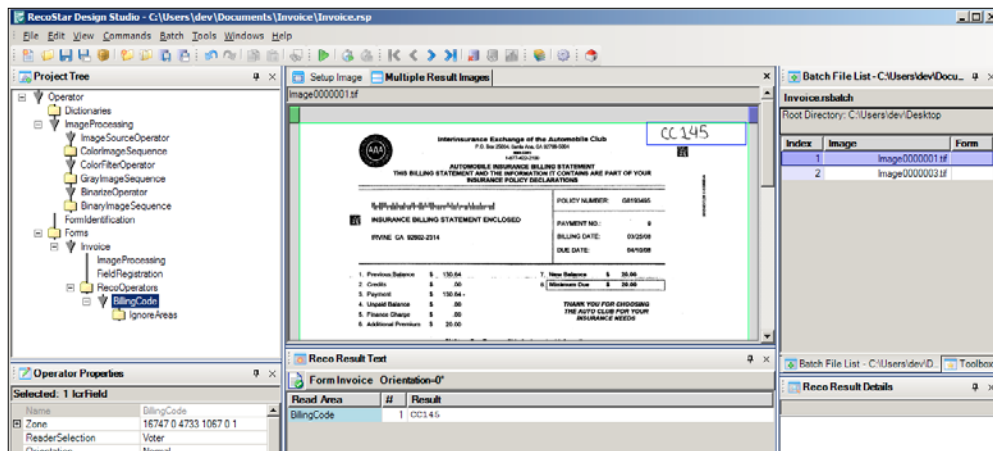


Since the billing code contains both letters and numbers, we will turn off the **LogicalContext** flag. There is a RecoStar tutorial available for Enterprise customers that explains more of these settings.



The layout configuration of the RecoStar field

Click on the Play button (the green triangle in the toolbar) to test the extraction. The results will be displayed in the **Reco Result Text** area:



Test results of the RecoStar extraction

Once RecoStar extraction is tested and is ready to be used in Ephesoft, save the project and copy the project file to the batch class RecoStar-extraction folder:
SharedFolders/BC99/fixed-form-extraction/



RecoStar project files have an .rsp extension.

Configuring Ephesoft to use the RecoStar project

The Invoice document type definition needs to be edited in order to use this RecoStar project. Go into the **Batch Class Management** tab of the administrative interface, and edit the Invoice document type. The new RecoStar project file should now appear in the list of project files under **First Page**, **Second Page**, **Third Page**, and **Last Page**. Select the project file as the first page to tell Ephesoft to use this RecoStar configuration when processing the first page of this document type. Create a new string field named **BillingCode**, click on **OK**, and then click on **Apply** to save changes.



Do not forget to verify that your document type name matches the RecoStar form name and that your field name matches the RecoStar field name.

You must also turn on RecoStar Extraction Switch in the RecoStar Extraction plugin within the Extraction module.




Document type fixed form project configuration

When you run a sample invoice through Ephesoft, you will see in validation that the value for Billing Code is extracted:

Validation with fixed form extraction results

Table extraction

Table extraction is used to extract repeating information from a form. On the following invoice, we might want to extract the date, hours, rates, and total from each row in the table.

 You must also turn on Table Extraction Switch in the Table Extraction plugin within the Extraction module.

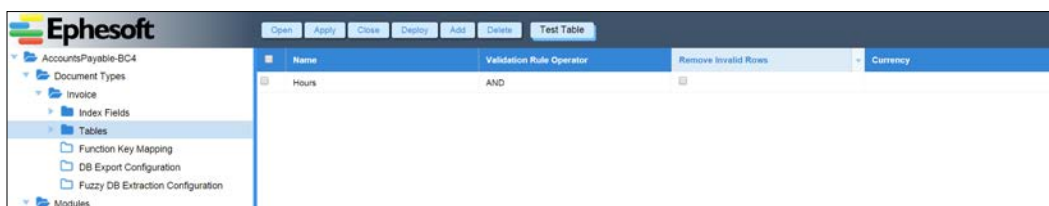
W/E	Employee / Description	Regular	Hours/Units	Rate	Net Amount
04-26-2009	Aguirre, Erica Packaging Clerk	Regular	7.00	12.75	89.25
04-26-2009	Cervantes, Michelle Clerk	Regular	7.00	12.75	89.25
04-26-2009	Chavez Zamudio, Rober Clerk	Regular	7.00	12.75	89.25
04-26-2009	MORENO, GLORIA Clerk	Regular	7.00	12.75	89.25
04-26-2009	Ponce, Sofia Packaging Clerk	Regular	7.00	12.75	89.25

Regular Hours	Over Time Hours	Other Hours	Total Hours
35.00	0.00	0.00	35.00
Regular Amount	Over Time Amount	Other Amount	Total Amount
446.25	0.00	0.00	446.25

A sample invoice with table information

From the administrative interface, edit the batch class, edit the Invoice document type, and then select **Tables**. Click on the **Add** button to define a new table. The following information can be specified about the table:

- **Name:** The name of the table.
- **Validation Rule Operator:** By default, all the validation rules must be satisfied in order for a table row to be valid. Select the **OR** validation rule if you wish the table row to be valid when any of the validation rules are satisfied.
- **Remove Invalid Rows:** If this is enabled, rows that fail to comply with the validation rules will not be extracted.
- **Currency:** This configuration setting indicates how columns that contain currency must be formatted.
- Enter the name and click on **Apply**.



New table extraction definition

Expand the **Tables** folder and navigate to **Table Columns**. Here, you will enter the following information about each column in the table:

- **Column Name:** The name of the column.
- **Description:** A description of the column.
- **Validation Pattern:** Values must match this regular expression to be considered valid.

- **Alternate Values:** When operators validate the content of this column, they may be presented a list of values to choose from. You can specify those values here. Use a semicolon (;) to separate the values.




Column Name	Description	Validation Pattern	Alternate Values
<input type="checkbox"/> Date	Date	d(2)-d(2)-d(4)	
<input type="checkbox"/> Hours	Hours	d(1,2);7;d(1,2)	
<input type="checkbox"/> Rate	Rate	d(1,2);7;d(1,2)	
<input type="checkbox"/> Total	Total	d(1,3);7;d(1,2)	

Table extraction column definition

Next, we will specify the extraction configuration for the tables for this invoice type. Select the **Table Extraction Rules** folder and click on **Add**. Add the following information for extracting information from this table:

- **Rule Name:** The name of the rule.
- **Start Pattern:** Regular expression pattern of a unique value or a word that is consistently at the beginning of the table. Ephesoft will not even try to extract tabular data until it sees this pattern in the document.
- **End Pattern:** Regular expression pattern of a unique value or a word that is consistently at the end of the table. When this pattern is encountered, Ephesoft will be certain that it has reached the end of the table. If the end pattern is not found, Ephesoft will examine lines of text all the way to the end of the document looking for table rows.
- **Table Extraction API:** This allows the administrator to choose any or all of the three different techniques for locating the content of a column. These techniques are as follows:
 - **Column Coordinates:** This allows the user to indicate the position on the page where the column begins and ends.
 - **Column Header:** This uses regular expressions to define the labels for the column headers. Ephesoft will assume that the content beneath a header is in that column.
 - **Regex Extraction:** This defines regular expressions for the content in the column as well as expressions for the content to the left and to the right.

[ Table extraction can span over multiple pages.]

In the preceding example, the start pattern is the word `branch` and the word `over` is the end pattern. We chose not to use the `Regular` or `Hours` pattern as the end pattern because both are used elsewhere on the page.

We will use **Column Header** and **Regex Extraction** APIs to locate and extract content from the columns of the table. Make sure that **Column Coordinates** is not checked and that **Column Header** and **Regex Extraction** are checked. Choose the **AND** operator from the dropdown list between the **Column Header** and the **Regex Extraction** checkboxes, as shown in the following screenshot:



Table extraction rule properties

Now that we have defined the table extraction rule, we need to define the column extraction rules. Expand the name of the table extraction rule that you created and navigate to the **Table Column Extraction Rules** folder. Here, you will be able to configure the following information for each column:

- **Column Pattern:** Used by regex extraction, this regular expression defines the format of the content in this column.
- **Between Left:** Used by regex extraction, this regular expression defines the pattern of text to the left of the value that you are extracting.
- **Between Right:** Used by regex extraction, this regular expression defines the pattern of text to the right of the value that you are extracting.
- **Column Header Pattern:** Used by column header extraction, this regular expression defines the pattern of the text in the column header.
- **Start Coordinate:** Used by column coordinate extraction, this property defines the coordinate that represents the left edge of the column. You can use the **Set Coordinates** button on this page to indicate on a sample image where the left edge is, and then, Ephesoft will set this value for you.
- **End Coordinate:** This is the same as **Start Coordinate** but defines the right edge of the column.
- **Multiline Anchor:** Used when extracting a table that has multiple lines of text per row. If this checkbox is selected, Ephesoft will start a new extraction row only when a value is found for this column.

- **Required:** Used to specify that a value is required in this column to create a valid row in the extracted table.
- **Extract Data From Column:** Ephesoft can perform regular expression extraction to extract a portion of the text from one of the other columns in this table. In order to accomplish this, select the column from which you wish to extract content.
- **Currency:** Indicates that the text in this column is currency and that validation should be performed based on the currency type specified in the table configuration.

Let's consider how we would configure Ephesoft to extract content from each of these columns.

- **Date:** This is simple, as it has a unique pattern; we can just specify the column pattern of `\d{2}-\d{2}-\d{4}`. It has a header pattern of `W\ /E`.
- **Hours:** The hours column has a pattern of `\d{2}\.\d{2}`, but this pattern is not unique to this column. To uniquely identify this column, we can also specify a between left pattern of `Regular`. This column has a header pattern of `Hours\ /Units`.
- **Rate:** The rate column has a pattern of `\d{2}\.\d{2}` and has a between left pattern of `\d{1,2}\.\d{2}` and a between right pattern of `\d{2}\.\d{2}`. It has a header pattern of `Rate`.
- **Total:** Finally, the total has a pattern of `\d{2}\.\d{2}` and a between left pattern of `\d{2}\.\d{2}`. It has a header pattern of `Net Amount`.

Once the values are entered, click on the **Validate Regex** button and then **Apply**.

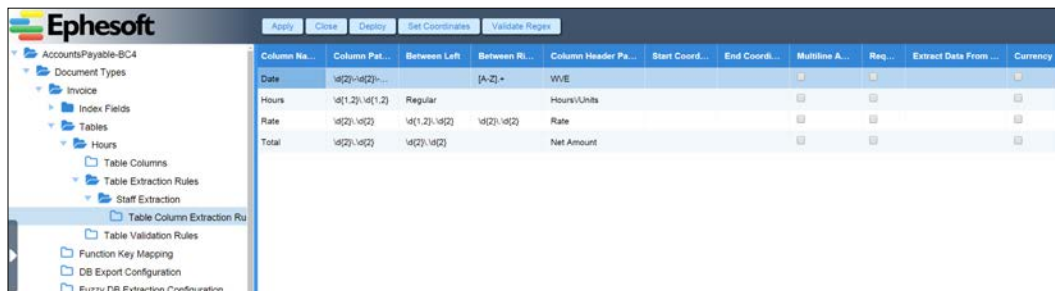
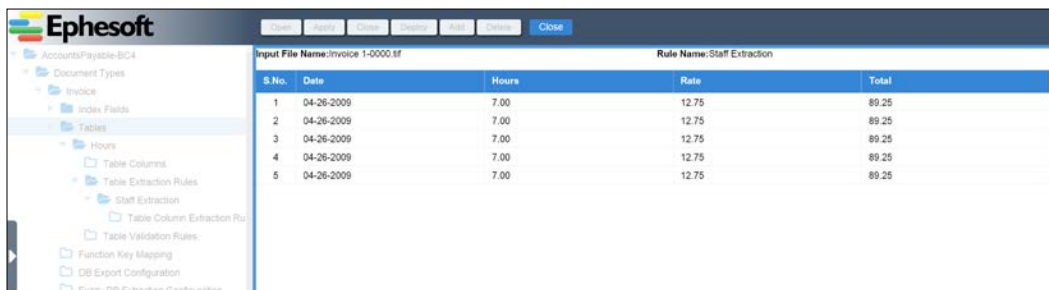


Table column extraction configuration

To test the rule, navigate back to the **Tables** folder. Drag a sample invoice to the drop area. Once the sample is uploaded, select the checkbox next to the **Hours** table and click on the **Test Table** button. You should see the extracted output in a screen, as follows:



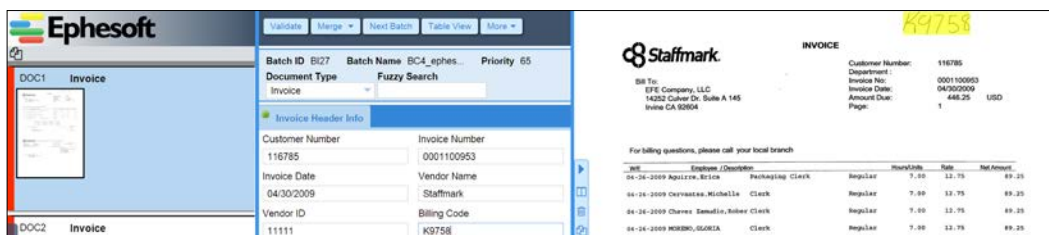
S.No.	Date	Hours	Rate	Total
1	04-26-2009	7.00	12.75	89.25
2	04-26-2009	7.00	12.75	89.25
3	04-26-2009	7.00	12.75	89.25
4	04-26-2009	7.00	12.75	89.25
5	04-26-2009	7.00	12.75	89.25

Table extraction test results



Repeating fields extracted using table extraction cannot be exported using the CMIS plugin; use scripting or an alternate export plugin to get tabular metadata into your document repository.

When indexing, operators can now click on the **Table View** button to view the table data:



Item	Employee / Description	Hours/Rate	Rate	Net Amount
04-26-2009 Aguirre, Brian	Packageing Clerk	Regular	7.00	12.75
04-26-2009 Contreras, Michella	Clerk	Regular	7.00	12.75
04-26-2009 Chavez, Robert	Clerk	Regular	7.00	12.75
04-26-2009 MENDOZA, GARCIA	Clerk	Regular	7.00	12.75

Displaying table view validation

Ephesoft displays the extracted table content. Operators can edit cells or edit and remove table rows to ensure that everything was captured properly:

Date	Hours	Rate	Total
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25

DATE	Employee / Description	Hourly Rate	Rate	Net Amount
04-26-2009	Aspines, Brian	Regular	12.75	89.25
04-26-2009	Aspines, Michelle	Clock	12.75	89.25
04-26-2009	Chaves, Robert	Clock	12.75	89.25
04-26-2009	MOORE, GOREA	Clock	12.75	89.25
04-26-2009	MOORE, Sofia	Perkaging Clock	12.75	89.25

Regular Hours	Over Time Hours	Other Hours	Total Hours
35.00	0.00	0.00	35.00
446.25	0.00	0.00	446.25

Subtotal	Total Amount Due
446.25	446.25

Table view validation

Scripting

Ephesoft can be configured to execute custom Java code in four places:

- In the workflow (at specific pre-defined points in Ephesoft's processing of a batch)
- When fields change in validation
- When an operator presses a function key
- Periodically at a configurable time

This code can manipulate the documents, populate field values, integrate with external systems, normalize data, and so forth.

The workflow scripts

The MailroomAutomationTemplate batch class, which we copied to create our custom batch, is configured with hooks to execute your own code in several places. For instance, if you want to perform a custom operation prior to validation, you can edit the following file: BC99/scripts/ScriptExtraction.java.

This file contains the implementation of a class named `ScriptExtraction`. Whenever Ephesoft's extraction module runs, it will call the `execute` method of this class:

```
public class ScriptExtraction implements IJDomScript {
    public Object execute(Document jdomObj, String methodName,
        String docIdentifier) {
        // execute custom code here
        return null;
    }
}
```

You do not need to compile your Java code or restart Ephesoft in order for any changes to the script to take effect. Ephesoft will detect any changes to the file prior to execution and re-compile the code automatically.

The following batch class scripts are available for customization:

- `ScriptAddNewTable.java`
- `ScriptAutomaticValidation.java`
- `ScriptDocumentAssembler.java`
- `ScriptExport.java`
- `ScriptExtraction.java`
- `ScriptFieldValueChange.java`
- `ScriptFunctionKey.java`
- `ScriptNewTableRowInsert.java`
- `ScriptPageProcessing.java`
- `ScriptValidation.java`



We recommend that you disable the default behavior of the `ScriptAutomaticValidation` script. The validations performed here are not included in `ScriptValidation`, resulting in inconsistent checking of these fields.

Study this script to learn how to manipulate the batch, and then disable all the functionality of the script. To best validate field values, you should create regular expressions in the validation rules; this is part of editing a document type.

Ephesoft persists the state of each batch instance to the file system as an XML file. When Ephesoft executes a script, it passes a JDOM document to the `execute` method. That document object is the result of parsing the XML file that represents the batch instance state. The author of a script can traverse this structure to determine metadata values, confidence levels, page counts, and so forth. Scripts can also change the state of a batch instance by modifying the JDOM object.



There is no documented schema for this XML structure. An Ephesoft batch XML file includes a `Document` element for each of the documents in the batch. A child element named `Identifier` specifies the ID of the document with which the `Document` element block is associated. Each `Document` element also includes a descendant `DocumentLevelField` element for each metadata field associated with the document. Various sub-elements of the `DocumentLevelField` element define the characteristics of the field.

Consider, for example, the table extraction performed in the previous section. Suppose that you want to add up the hours from each row of the invoice and store the total in a metadata field named `TotalHours`. Here is what the batch instance state looks like when persisted to XML:

```
<Batch>
  <Documents>
    <Document>
      <Identifier>DOC1</Identifier>
      <DocumentLevelFields>
        <DocumentLevelField>
          <Name>TotalHours</Name>
          <Value>0.0</Value>
        </DocumentLevelField>
      </DocumentLevelFields>
      <DataTables>
        <DataTable>
          <Name>LaborTable</Name>
          <Rows>
            <Row>
              <Columns>
                <Column><Value>04-26-2009</Value></Column>
                <Column><Value>12.75</Value></Column>
                <Column><Value>7.00</Value></Column>
              </Columns>
            </Row>
            <!-- more rows here -->
          </Rows>
        </DataTable>
      </DataTables>
    </Document>
  </Documents>
</Batch>
```

```

        </Rows>
    </DataTable>
</DataTables>
</Document>
</Documents>
</Batch>

```

The actual XML file contains significantly more information; we have reduced the content to just the elements that are relevant to this example. Here is an example of some code that would sum up all the hours in the `Hours` cells and write the total into the `TotalHours` field:

```

public Object execute(Document jdomObj, String methodName,
                      String docId)
{
    Exception exception = null;
    try {

        // for each invoice document
        String docPathStr =
            "/Batch/Documents/Document [Type='Invoice']";
        XPath docXPath = XPath.newInstance(docPathStr);
        List<Element> docList = docXPath.selectNodes(jdomObj);
        for (Element doc : docList) {

            // for each "hours" cell in the hours table
            String valuePathStr =
                "DataTables/DataTable [Name='Hours']" +
                "/Rows/Row/Columns/Column [2] /Value";
            XPath valueXPath = XPath.newInstance(valuePathStr);
            double totalHours = 0;
            List<Element> hoursList = valueXPath.selectNodes(doc);
            for (Element hoursElement: hoursList) {

                // add the hours to the total
                totalHours += Double.parseDouble(hoursElement.getText());
            }

            // write the total hours into a metadata field
            String ttlString =
                "DocumentLevelFields/" +
                "DocumentLevelField [Name='TotalHours'] /Value";
            XPath ttlXPath = XPath.newInstance(ttlString);
            Element ttlElem = (Element) ttlXPath.selectSingleNode(doc);

```

```


        ttlElem.setText(String.valueOf(totalHours));
    }
}
catch (JDOMException e) {
    System.err.println("script failed: " + e);
    e.printStackTrace();
    exception = e;
}

// return null for success
return exception;
}

```

This script first iterates across all the documents of the Invoice type. Then, it finds the Hours cell for each row and adds its value to a running total. Next, it finds the TotalHours field and sets the field's value to be this total.

This script has also been simplified; in practice, you will discover that Ephesoft does not provide a Value tag for empty fields. Your script will need to call JDOM to add the Value tag to the TotalHours field before setting its value.


 Scripts used in validation such as ScriptFunctionKey.java or ScriptValidation.java can update the ErrorMessage tag in the batch XML to display error messages to the user on the screen.

The result is that if the Hours values have been successfully extracted from the table, the **Total Hours** field is populated with their sum.

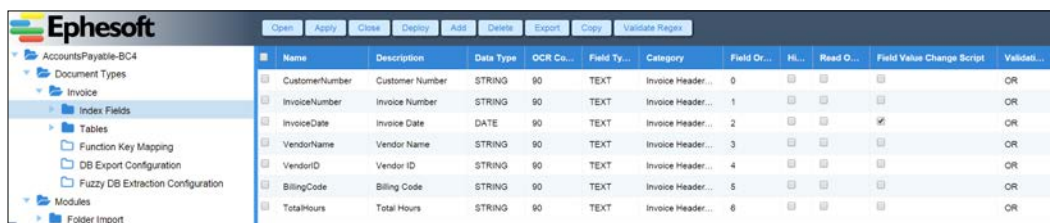


Validation displaying calculated field from extraction script

Triggering scripts when a field is edited

Ephesoft can call a script when a change is made to the value of specified fields on the validation screen. As an example, we will implement a check to see whether the field is of the `DATE` type, and if so, change any date formatted with - to /. The script will run when the value is changed in the text field and the focus changes to another field, the document is saved (*Ctrl + q* or *Ctrl + e*), or another document is selected.


In the batch class administration area's list of index fields, you will find a **Field Value Change Script** checkbox associated with each field. Use this checkbox to indicate that the **InvoiceDate** field should trigger the script when its value is changed. Click on **Apply** when you have made the change.



Name	Description	Data Type	OCR Co.	Field Ty.	Category	Field Or.	H.	Read O.	Field Value Change Script	Validat...
CustomerNumber	Customer Number	STRING	90	TEXT	Invoice Header...	0			<input type="checkbox"/>	OR
InvoiceNumber	Invoice Number	STRING	90	TEXT	Invoice Header...	1			<input type="checkbox"/>	OR
InvoiceDate	Invoice Date	DATE	90	TEXT	Invoice Header...	2			<input checked="" type="checkbox"/>	OR
VendorName	Vendor Name	STRING	90	TEXT	Invoice Header...	3			<input type="checkbox"/>	OR
VendorID	Vendor ID	STRING	90	TEXT	Invoice Header...	4			<input type="checkbox"/>	OR
BillingCode	Billing Code	STRING	90	TEXT	Invoice Header...	5			<input type="checkbox"/>	OR
TotalHours	Total Hours	STRING	90	TEXT	Invoice Header...	6			<input type="checkbox"/>	OR

Field value change script configuration

To use this feature, you must first turn on the **Field Value Change Script** switch in the Validate Document plugin within the Validate Document module.

 This feature, when enabled, results in communication between the web browser and Ephesoft whenever a field value is edited. Only deployments where the client has a high-speed connection to the server should make use of this feature.

Now, the `changeField` method within `ScriptFieldValueChange.java` will be invoked whenever an operator changes the value of the `InvoiceDate` field. Here is an implementation of `changeField` that reformats date fields with slashes instead of dashes:

```
public void changeField(Document doc, String fieldName,
                        String docIdentifier)
{
    try {

        // find the field element that the operator is editing
        String fieldPathStr =
            "/Batch/Documents/Document[Identifier = '" +
            docIdentifier +
```

```
        "']/DocumentLevelFields/DocumentLevelField[Name = '" +
        fieldName +
        "']";

XPath fieldPath = XPath.newInstance(fieldPathStr);
Element field = (Element) fieldPath.selectSingleNode(doc);

// if this is a date
XPath typePath = XPath.newInstance("Type");
Element type = (Element) typePath.selectSingleNode(field);
if ("DATE".equals(type.getValue())) {

    // get the value
    XPath valuePath = XPath.newInstance("Value");
    Element value = (Element) valuePath.selectSingleNode(field);
    String oldValue = value.getValue();

    // reformat the date
    SimpleDateFormat oldFormat =
    new SimpleDateFormat("MM-dd-yyyy");
    SimpleDateFormat newFormat =
    new SimpleDateFormat("MM/dd/yyyy");
    Date date = oldFormat.parse(oldValue.trim());
    String newValue = newFormat.format(date);

    // persist the reformatted value
    value.setText(newValue);
}
}
catch (JDOMException x) {
x.printStackTrace();
}
catch (ParseException x) {
// format of date is not MM-dd-yyyy; ok to ignore
}
}
```

With this script in place, the following document's invoice date was modified by an operator to be in a date format with - as the separator:

The screenshot shows the Ephesoft validation screen. On the left, the 'Invoice' tab is selected, displaying fields for Customer Number (116785), Invoice Number (0001100953), Invoice Date (04-30-2009), Vendor Name (Staffmark), Vendor ID (11111), Billing Code (K9758), and Total Hours (35.0). On the right, the 'INVOICE' section shows the Staffmark logo, customer information, and a table of billing details. The invoice date is 04/30/2009, and the amount due is 446.25 USD.

DATE	Employee / Description	Rate	Hours	Net Amount
04-30-2009	Aguiar, Rita	Regular	7.00	12.75
04-30-2009	Gervaseau, Michelle	Clerk	7.00	12.75
04-30-2009	Charles, Sandra, Robert	Clerk	7.00	12.75
04-30-2009	MORRIS, GLORIA	Clerk	7.00	12.75
04-30-2009	Rouse, Sofia	Package Clerk	7.00	12.75
Regular Hours		Over Time Hours	Other Hours	Total Hours
35.00		0.00	0.00	35.00
Regular Amount		Over Time Amount	Other Amount	Total Amount
446.25		0.00	0.00	446.25

Validation screen with manually entered invoice date

When the operator selects another field, Ephesoft indicates briefly that a script is running. Soon thereafter, the format of the date changes to contain / rather than - in the invoice date.

The screenshot shows the Ephesoft validation screen, similar to the previous one, but with the invoice date formatted as 04/30/2009. The rest of the screen, including the invoice details and the table of billing details, remains the same.

Validation screen with formatted invoice date from a change script

Triggering scripts from a function key

Ephesoft can call a script when a function key (*F1-F11*, excluding *F5*) is pressed. The function key mapping can be created for each document type. To create a function key mapping, first select a document and go to the **Edit** screen. When you configure this mapping, Ephesoft will also add a submenu item under the **More** dropdown in the validation area of the operator interface that you can click with a mouse to perform the same function. There is a **Function Key Mapping** section in the batch class administration area for each batch class. Navigate to this area for the document type for which you want to add a function key behavior. Click on the **Add** button.

Enter `myMethod` for the method name (the name of a Java method in the `ScriptFunctionKey.java` file) and any method description (this is displayed in the validation interface), and select `F2` as the shortcut key that should be pressed to trigger this script (`F1-F11` excluding `F5`).



You may not want to use `F1` to trigger your script, since some browsers use that shortcut for help.

Click on the **Apply** button to save the configuration.



The function key script configuration

When the `F2` key is pressed in the review or validation screen, the following method will be executed. For a function key method, Ephesoft passes the document identifier of the document selected by the operator and the DOM object of the batch XML document. This method uses the document identifier to locate the document that the operator has selected. Once the document is found, this method totals the hours in the Labor table and saves the results in the **TotalHours** field (just as in the earlier example):

```
@SuppressWarnings("unchecked")
public void myMethod(String docId, Document jdomObj) {
    try {
        // path to invoice documents
        String docPathStr =
            "/Batch/Documents/Document[Identifier = '" + docId + "']";
        XPath documentsPath = XPath.newInstance(docPathStr);

        List<Element> docs = documentsPath.selectNodes(jdomObj);
        for (Element doc : docs) {

            // for each "hours" cell in the labor table
            String hoursPathStr =
                "DataTables/DataTable[Name='Hours'] " +
                "/Rows/Row/Columns/Column[2]/Value";
            XPath valuePath = XPath.newInstance(hoursPathStr);
```

```

List<Element> hoursList = valuePath.selectNodes(doc);
double totalHours = 0;
for (Element hoursElement: hoursList) {

    // add the hours to the total
    totalHours += Double.parseDouble(hoursElement.getText());
}

// write the total hours into a metadata field
String totalPathStr =
    "DocumentLevelFields/" +
    "DocumentLevelField[Name='TotalHours']/Value";
XPath totalPath = XPath.newInstance(totalPathStr);

Element total = (Element) totalPath.selectSingleNode(doc);

total.setText(String.valueOf(totalHours));
}
}
catch (JDOMException e) {
    System.err.println("script failed: " + e);
    e.printStackTrace();
}
}

```

Now, looking at the operator's interface, you can see an *F2* item under the **More** dropdown in the **Function Keys** submenu. Click on the table icon to view the table extraction value for the **Labor** table.

In the table results for the validation, we can add a new line by pressing the **Insert** button. Enter 8.0 in the **Hours** column and then click on the **Back** button.

Date	Hours	Rate	Total
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25
04-26-2009	7.00	12.75	89.25
04-26-2009	8.00	12.75	102.00

REG	Employee / Description	Hourly Rate	Reg	Net Amount
04-26-2009	Aguiar, Eric	Regular	7.00	89.25
04-26-2009	Cervantes, Michelle	Clerk	7.00	89.25
04-26-2009	Chaves, Sandra/Robert	Clerk	7.00	89.25
04-26-2009	MORERO, GORDIA	Clerk	7.00	89.25
04-26-2009	Rouse, Sofia	Packaging Clerk	7.00	89.25

Regular Hours	Over Time Hours	Other Hours	Total Hours
35.00	0.00	0.00	35.00
Regular Amount	Over Time Amount	Other Amount	Total Amount
446.25	0.00	0.00	446.25

SUBTOTAL: 446.25

Table view validation with manually entered row

Validation screen displaying total hours after pressing the F2 key

There is a single application script for each Ephesoft server that is executed periodically at a configurable interval. This script can be used for any task that you would like at the application and not the batch level. For instance, it could be used for tasks such as creating nightly reports or pulling batches from multiple locations into a single batch class-monitored folder.

Whenever Ephesoft runs the application script, it will call the `execute` method in this file:

The property that sets the interval is `dcma.applicationLevelScript.cronExpression`.

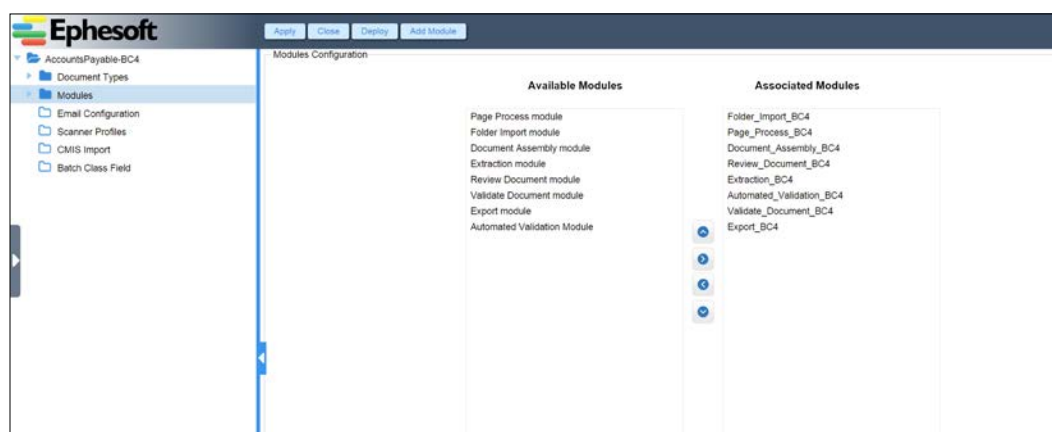
The custom workflow

Ephesoft comes with several batch classes, each of which implements a workflow that is of general use. You can solve many business problems simply by copying one of the pre-configured batch classes and using the associated workflow. Sometimes, however, you need more steps (to accommodate the specific needs of your organization) or fewer steps (to accelerate performance).

Customizing batch class workflows

Every batch class has its own capture workflow that includes modules and plugins. Plugins are steps in the capture workflow and modules are used to group plugins to manage batches.

The following screenshot illustrates how administrators can add or remove modules from individual batch classes. Modules are simply groups of plugins; administrators can create modules by clicking on the **Add Module** button from the batch class administration area. After the module is given a name and a description, it will appear in the list of available modules. You can use the arrow buttons to move the module into the list of modules associated with your batch class and adjust the order in which the module will execute within this workflow:

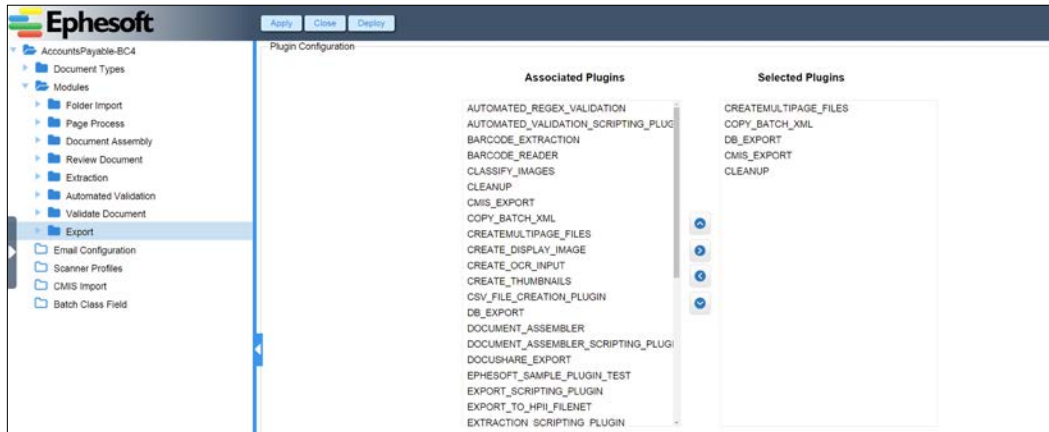


A batch class module configuration

Administrators can add and remove plugins from individual modules by clicking on the module's name in the navigation tree on the left of the batch class administration area. All the plugins installed in Ephesoft are displayed on the left-hand side as **Associated Plugins** and administrators can assign the desired plugins to the module by using the arrow button to move it to the list of **Selected Plugins**.



When you click on a plugin from the list of associated plugins, Ephesoft will highlight the plugins that the clicked plugin depends on.



Batch class module plugin configuration

Once the administrator is satisfied with the module and plugin configuration, he/she is required to deploy the new capture workflow. Click on the **Apply** button and then on the **Deploy** button. Ephesoft will check all the dependencies for the selected plugins and deploy the workflow. You will receive a success message if all dependencies are met and the workflow is deployed.

Writing a custom plugin

Ephesoft allows developers to write their own plugins and install them to Ephesoft. Developers should prepare the following files to install their custom plugins to Ephesoft:

- A .jar file containing code that will be executed by the workflow and the Spring application context XML file
- An XML file containing the plugin configuration including plugin name, author, dependencies, and version information

These two files are bundled into a ZIP archive so that administrators can upload and install the plugin to Ephesoft as described earlier.

The following steps are designed to create a sample plugin that sets a batch-level property. The name of the property and the value to be assigned are configurable as the plugin parameters.

We recommend that you use Maven to build your plugin, as we do in this example, but it is not required in order to create a plugin:

1. Create a Maven project. Enter artifact information as follows:
 - **Group ID:** `com.mycorp`
 - **Artifact Id:** `ephesoft-sample-plugin`
 - **Version:** `1.0.0`
 - **Packaging:** `jar`
2. Create the `ephesoft-sample-plugin/src/main/java/com/mycorp / SamplePlugin.java` file:

```
package com.mycorp;

// import statements here

public class SamplePlugin {

    private static final String
    PLUGIN_NAME = "EPHESoft_SAMPLE_PLUGIN";

    // ephesoft service to interact with batch
    @Autowired
    private BatchSchemaService batchSchemaService;

    // ephesoft service to access plugin properties
    @Autowired
    @Qualifier("batchInstancePluginPropertiesService")
    private PluginPropertiesService props;

    // code executed before plugin
    @PreProcess
    public void preProcess(final BatchInstanceID biid, String
    workflow)
    {
        Assert.notNull(biid);

        // save the state of the batch prior to execution
        BackUpFileService.backUpBatch(biid.getID());
    }

    // code executed after plugin
    @PostProcess
```

```
public void postProcess(final BatchInstanceID biid, String
workflow)
{
    Assert.notNull(biid);

    // save the state of the batch after execution
    BackUpFileService.backUpBatch(biid.getID(), workflow);
}

// entry point for plugin
public void sampleMethod(BatchInstanceID biid, String
workflow)
    throws Exception
{
    String biID = biid.getID();

    // create the batch level field
    BatchLevelField blf = createBatchLevelField(biid);
    Batch batch = batchSchemaService.getBatch(biid);

    // find of create the list of batch level fields
    BatchLevelFields blfs = batch.getBatchLevelFields();
    if(blfs == null) {
        blfs = new BatchLevelFields();
    }
    blfs.getBatchLevelField().add(blf);

    // add the new batch level field to the list
    batch.setBatchLevelFields(blfs);
    batchSchemaService.updateBatch(batch);
}

// helper method to create batch level field
// using info from batch properties
private BatchLevelField createBatchLevelField(String biid)
{
    BatchLevelField blf = new BatchLevelField();
    String name = props.getPropertyValue(biid, PLUGIN_NAME,
"sample_plugin.name");
    String value = props.getPropertyValue(biid, PLUGIN_NAME,
"sample_plugin.value");
    String type = props.getPropertyValue(biid, PLUGIN_NAME,
"sample_plugin.type");
    blf.setName(name);
}
```

```

    blf.setValue(value);
    blf.setType(type);
    return blf;
}

```

3. Create the `ephesoft-sample-plugin/src/main/resources/META-INF/sample-plugin-context.xml` file. You will need to include several Spring namespace attributes on the bean tag that have been omitted from this example for readability:

```

<?xml version="1.0" encoding="UTF-8"?>
<beans default-autowire="byName">
<import resource=
    "classpath:/META-INF/applicationContext-batch.xml" />
<import resource=
    "classpath:/META-INF/applicationContext-data-access.xml"
/>

<bean id="myPlugin" class="com.mycorp.SamplePlugin" />

</beans>

```

4. Add `ephesoft.jar` to your project in a `lib` subfolder of the project's root folder. This JAR file can be found in `Application\WEB-INF\lib`.
5. Add a dependency in your `pom.xml` file for `ephesoft.jar`. The correct way to accomplish this is to install the JAR file into your local repository, but for the purpose of this example, you can simply create a system-scoped dependency by adding the following to your dependencies section:

```

<dependency>
  <groupId>com.ephesoft.dcma</groupId>
  <artifactId>ephesoft</artifactId>
  <type>jar</type>
  <scope>system</scope>
  <systemPath>${basedir}/lib/ephesoft.jar</systemPath>
</dependency>

```

6. Use the `mvn package` command to create the JAR file.
7. Create a plugin configuration file called `sample-plugin-config.xml`.

```

<?xml version="1.0" encoding="UTF-8"?>
<plugin>
<jar-name>ephesoft-sample-plugin-1.0.0.jar</jar-name>
  <plugin-name>EPHESOFT_SAMPLE_PLUGIN</plugin-name>
  <plugin-desc>Ephesoft Sample Plugin</plugin-desc>

```

```
<plugin-workflow-name>EPHESOFT_SAMPLE_PLUGIN</plugin-  
workflow-name>  
<plugin-service-instance>myPlugin</plugin-service-instance>  
<method-name>sampleMethod</method-name>  
<is-scripting>FALSE</is-scripting>  
<back-up-file-name>N/A</back-up-file-name>  
<script-name>N/A</script-name>  
<application-context-path>sample-plugin-  
context.xml</application-context-path>  
<plugin-properties>  
<plugin-property>  
  <name>sample_plugin.name</name>  
  <type>STRING</type>  
  <description>Field Name</description>  
  <is-mandatory>FALSE</is-mandatory>  
  <is-multivalue>FALSE</is-multivalue>  
</plugin-property>  
<plugin-property>  
  <name>sample_plugin.value</name>  
  <type>STRING</type>  
  <description>Field Value</description>  
  <is-mandatory>FALSE</is-mandatory>  
  <is-multivalue>FALSE</is-multivalue>  
</plugin-property>  
<plugin-property>  
  <name>sample_plugin.type</name>  
  <type>STRING</type>  
  <description>Field Type</description>  
  <is-mandatory>TRUE</is-mandatory>  
  <is-multivalue>FALSE</is-multivalue>  
  <sample-values>  
    <sample-value>STRING</sample-value>  
  </sample-values>  
  <sample-values>  
    <sample-value>INTEGER</sample-value>  
  </sample-values>  
</plugin-property>  
</plugin-properties>  
<dependencies>  
  <dependency>  
    <type-of-dependency>Order Before</type-of-dependency>  
    <dependency-value>IMPORT_BATCH_FOLDER/IMPORT_MULTIPAGE_FILES</  
dependency-value>  
  </dependency>  
</dependencies>  
</plugin>
```

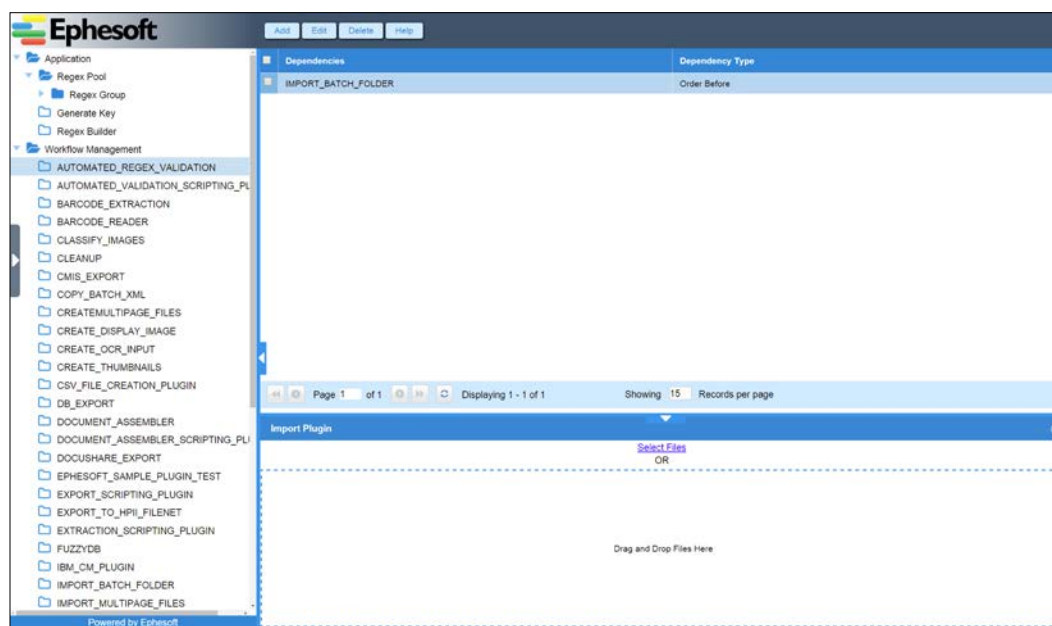
8. Zip up the configuration file and the JAR file into a `my-plugin.zip` file. You should now be able to install this zipped-up plugin into Ephesoft.

The plugin configuration file should comply with the following rules:

- The `application-context-path` tag must refer to the Spring context file that you created. This path is relative to the `META-INF` folder within your JAR.
- The `plugin-service-instance` tag should match the ID of the bean defined in the Spring configuration file.
- The `method-name` tag must match the method of your bean that you want Ephesoft to invoke.

Adding a plugin to Ephesoft

Now, you can add your plugin to Ephesoft from the **System Configuration** area by navigating to **Workflow Management**. Administrators can view all the available plugins and add new plugins to the system by uploading a ZIP file that contains all the files necessary to run a plugin for Ephesoft.



The workflow management user interface

You can import plugins by dragging them to the area at the bottom of the **Workflow Management** screen or by clicking on the **Select Files** button. Plugins may be dependent on other plugins. You can manage these dependencies by selecting a plugin and using the **Add**, **Edit**, and **Delete** buttons at the top of the screen.

This plugin can now be added to batch class workflows as described in the previous section.

The Web Services API

In software, a web service is a way for applications to communicate over a network. Client applications send requests over HTTP to server applications that typically respond with XML or JSON.

Ephesoft provides web services that allow external clients to execute both document processing and administrative actions. It is also possible to simply start a batch of documents processing through the system via a web service; this is how Ephesoft implemented its SnapDoc mobile application.

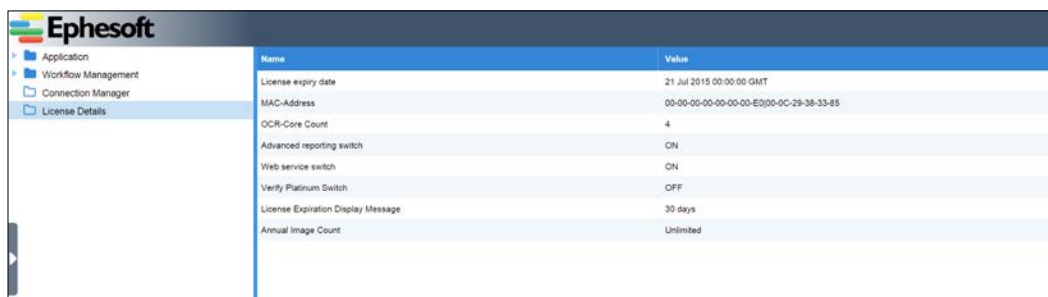
Ephesoft's Web Services API allows you to create solution architectures that break the pipeline paradigm. Traditionally, capture solutions were designed with scanning solutions flowing into Ephesoft, which then exported the indexed content into a document repository. Now, Ephesoft can be accessed as a service from one or more systems in your enterprise.

For example, your organization may have a portal through which external users can upload invoices. Ephesoft has to process these invoices, but the end users do not need (or even want) the full Ephesoft operator UI. Instead, it makes more sense for the portal to invoke Ephesoft's web services to extract information from the invoices.

Let us consider the process of integrating Ephesoft into an existing portal application by using the Web Services API.

Enabling the Web Services API

Before you use the web services, you must first make sure that your license allows the use of this feature. An administrator can verify this by navigating to the **License Details** section of the **System Configuration** area. The **Web service switch** property will indicate whether or not this feature is enabled:



Name	Value
License expiry date	21 Jul 2015 00:00:00 GMT
MAC-Address	00-00-00-00-00-00-E000-0C-29-38-33-85
OCR-Core Count	4
Advanced reporting switch	ON
Web service switch	ON
Verify Platinum Switch	OFF
License Expiration Display Message	30 days
Annual Image Count	Unlimited

License details with web service turned on

The available web services

The available web services cover a wide array of functions, from administrative tasks such as creating a document type, to capture tasks such as classifying TIFFs, running key value extraction, and generating searchable PDFs. Specific services include the following:

- **Create Searchable PDF:** This generates a searchable PDF from a TIFF
- **Get Batch Instance List:** This returns a list of all running batch instances of a specified status
- **Extract KV for Document Type:** This performs key value extraction for a specified document type within a batch class

There is another set of web services developed for use by the SnapDoc mobile application but which you can use if you have purchased a license that provides access to the web services. These Mobile Web Services include the following:

- **OCR Classify Extract:** This calls the Classification and Extraction plugins against a PDF or ZIP file of single-page TIFF files
- **Execute Mobile Upload:** This pushes PDFs to Ephesoft to be processed by an existing batch class

Mobile Web Services are documented separately and require an additional step to enable. Uncomment the following line from `Application\applicationContext.xml`:

```
<import resource="classpath:/META-INF/applicationContext-  
mobilewebservice.xml" />
```

Invoice processing portal example

The example that we will use for demonstrating Ephesoft's Web Services API is to enhance an existing portal to support the following new features:

- Allow customers to upload invoices in the PDF format
- Automatically extract metadata from the invoice
- Allow the customer to verify the extracted data

We need to provide an interface that will not confuse a customer who is using it for the first time with no training. Rather than hand-code all of the client-side JavaScript, we will use the AngularJS library.

Ephesoft provides several services that support extraction:

- Extract KV
- Extract Fixed Form
- OCR Classify Extract

Extract KV does not OCR the content, so we have to call another web service first. This is inefficient, so we will not use this service. Extract Fixed Form will OCR the content, but we cannot guarantee the layout of our invoices, so fixed form extraction is not the right choice for this example. We will use OCR Classify Extract; this will perform both OCR and any extraction (not just fixed form). This is perfect for this example because we use both key value and FuzzyDB extraction in the processing of our invoices. Additionally, the OCR Classify Extract service allows you to specify the document type. Since we already know that we will be processing invoices, we can skip the classification step, thus speeding up the processing of our documents.

Creating the upload view

We know we need a web page that accepts documents for upload, provides a button to extract content, and displays the results of the extraction.

We can begin by creating a view to allow users to upload their invoice. Therefore, we create the following page:

```
<div class="jumbotron">
  <h1>Invoice Processing</h1>
  <p class="lead">Please upload an invoice for processing.</p>
  <div class="row">
    <input type="file" id="file" name="file" file="file"/>
    <button ng-click="extract()">Extract</button>
  </div>
</div>

<div class="row">
  <div ng-repeat="dlf in docLevelFields">
    <div class="input-group">
      <span class="input-group-addon" id="basic-addon1">{{
        dlf.Name }}</span>
      <input type="text" class="form-control" placeholder="123.45"
        value="{{
          dlf.Value }}" aria-describedby="basic-addon1">
    </div>
    <br />
  </div>
</div>
```

This HTML file sets up the file upload control, an **Extract** button, and an area to show the resulting fields from Ephesoft. We use the AngularJS library to communicate with the server. The `ng-click` directive calls an `extract` function that we will define in the controller file. The `docLevelFields` variable will be populated in this controller as well.

Creating the controller and calling the web services

Next, we create a controller to call Ephesoft via the Web Services API and bind the results to the `docLevelFields` variable in the view:

```
angular.module('invoiceProcessingPortalApp')
.directive('file', function() {
  return {
    scope: {
      file: '='
    },
    link: function(scope, el, attrs) {
      el.bind('change', function(event) {
        var file = event.target.files[0];
        scope.file = file ? file : undefined;
        scope.$apply();
      });
    }
  };
});

.controller('MainCtrl', function ($scope, $http, x2js) {

  $scope.extract = function(){

    var req = {
      method: 'POST',
      url: 'http://mycorp.com/dcma/rest/ocrClassifyExtract',
      headers: {
        'Authorization': 'Basic ZXBoZXNvZnQ6ZGVtbw==',
        'Content-Type': undefined
      },
      transformRequest: function(data, headersGetter) {
        var formData = new FormData();
        angular.forEach(data, function(value, key) {
          formData.append(key, value);
        });
        var headers = headersGetter();
        delete headers['Content-Type'];
        return formData;
      },
      transformResponse: function(value) {
        var valueJson = x2js.xml_str2json(value)
        return valueJson;
      }
    };
  }
});
```

```

    }
  };

  req.data = {};
  req.data['BatchClassIdentifier'] = 'BC6';
  req.data['docType'] = 'Invoice';
  req.data[$scope.file.name] = $scope.file;

  $http(req)
    .success(function(data, status, headers, config) {
      $scope.docLevelFields =
        data.Web_Service_Result.Result.Batch.Documents
        .Document.DocumentLevelFields.DocumentLevelField;
    })
    .error(function(data, status, headers, config) {
      $scope.message = data;
    });
  }
}
);

```

The first part of the controller, the directive, places the contents of the file upload input into the `$scope.file` variable.

Next, we implement the `extract` function that is called when the **Extract** button is clicked. In order to call the web service, we must first build the request. By looking at Ephesoft's documentation for the OCR Classify Extract web service, we learn that we must POST our request to the `/dcma/rest/ocrClassifyExtract` endpoint. Our request data consists of a few fields, and we specify `BatchClassIdentifier` as `BC99` (this is the batch class that processes our invoices). We define a property where the name of the property is the content filename and the value is the file content. We set `docType` to `Invoice` to skip the classification plugin, which speeds up the processing of invoices.

We use the `x2js` library to transform the response from XML to JSON.

Finally, in the success block of our code, we place `documentLevelFields` from the response into `$scope.docLevelFields` to be used within the view.



When calling the Ephesoft Web Service API from a browser, CORS will need to be enabled on the Ephesoft server. See the online Ephesoft documentation for instructions.

Using the app

Once the app is up and running, you can use the **Upload File** button to select your document, and then click on **Extract**. Your results should look something like this:

The screenshot shows a web application titled "Invoice Processing Portal". The main heading is "Invoice Processing" with the instruction "Please upload an invoice for processing." Below this, there is a "Choose File" button and a text input field containing "InvoiceB.pdf". To the right of the input field is an "Extract" button. Below the upload section, there are several input fields for invoice data: "VendorName" (containing "Staffmark"), "CustomerNumber" (containing "116785"), "TotalHours" (containing "63.75"), "InvoiceDate" (containing "04/30/2009"), "VendorID" (containing "55555"), "InvoiceNumber" (containing "0001100953"), and "BillingCode" (containing "123.45"). At the bottom of the form, there is a footer with the text "Intelligent Document Capture with Ephesoft" on the left and "Zia Consulting © 2015" on the right.

A sample web application using Ephesoft web services

You will notice that all the document-level fields that you have configured Ephesoft to extract are displayed. This is obviously just the first step to creating a fully functioning web application; we will probably need to push the results of the extraction to another system, possibly an ERP or content management system to be processed further.

Summary

Congratulations! You can conquer almost any document capture task with the tools and skills now at your disposal. You should be able to configure Ephesoft to implement the capture needs of most organizations, and you know how to customize and extend Ephesoft to handle anything that it can't do *out of the box*.

In the next chapter we will share some helpful tips including troubleshooting, administration, Active Directory/LDAP setup, and e-mail processing configuration.

5 Tips

Now that you have a thorough understanding of the Ephesoft system, it is time to turn our attention to some items that will prove helpful during the implementation and support of your system. In this chapter, we will provide helpful tips including the following:

- Troubleshooting
- Restarting a batch
- No blank forms available for training
- Setting up Active Directory
- Setting up e-mail processing

Troubleshooting

In this section, we explain common troubleshooting methods for Ephesoft.

Logging

The primary log file for Ephesoft is located at `Ephesoft\Application\logs\dcma-all.log`.

Sometimes, additional information can be found in these files:

- `Ephesoft\Application\logs\dcma_report_all.log`
- `Ephesoft\Application\logs\dcma-user.log`
- `Ephesoft\JavaAppServer\logs\catalina.*.log`
- `Ephesoft\JavaAppServer\logs\stdout_*.log`
- `Ephesoft\JavaAppServer\logs\stderr_*.log`

Ephesoft can be configured to log in greater detail by editing this file, `Ephesoft\Application\log4j.xml`. Change the level from `WARN` to `INFO`, or for the maximum amount of information, `DEBUG`:

```
<logger name="com.ephesoft">
  <level value="DEBUG" />
</logger>
```

Monitoring batch progress

The Batch Instance Management area of the administration interface shows the status of each batch, but without much detail. The status is not updated automatically, and so the list has to be refreshed manually.

Batch processing can also be monitored by configuring logging to the `INFO` level and watching the logs. If you are using Linux, you can *tail* the file. On Windows, you can install Cygwin to get access to the `tail` command (as well as a number of other convenient Unix command-line tools) or you can use an application that performs the same function, such as BareTail. These can be found online at <http://www.baremetalsoft.com/baretail/> and <http://www.cygwin.com/>.

The third way of monitoring batch progress is to look in the batch instance folder. First, find the batch's batch instance ID in the Batch Instance Management area. This will begin with `BI` and end with a hexadecimal number. Then, watch Ephesoft's working folder for this batch. If the batch ID is `BI99`, Ephesoft will write working files to this folder, `SharedFolders\ephesoft-system-folder\BI99`.

Sort the display to show the most recently modified files first. This will provide a good sense of what Ephesoft is doing.

Examining the batch file

The workflows persist workflow data (assembly information, extracted index field values, and so on) in an XML file for each batch. Each module creates a snapshot of this data when it is executed. Look at the XML data to help debug assembly and extraction by looking at values, confidence levels, and alternative values.

The batch files are stored in the batch instance working folder and cleaned up when the batch completes successfully or is deleted.

Restarting the batch

Failed batch instances can be restarted from an earlier step in the workflow by using the administrative Batch Instance Management interface. This is faster than having to create a new batch instance. One or more batch instances can be selected and restarted from the beginning of a previous or current module.

For example, if a modification is made to a script in the workflow, restarting means that a new workflow does not need to be started to test these changes.

No blank forms available for training

Classification is the most accurate when the system is trained with blank forms (a form that has not been completed). If blank forms are not available, accurate classification can still be achieved.

The first option is to redact (remove sensitive and instance-unique data) on the samples you have before uploading them to Ephesoft for training.

The second option involves editing the HOOCR file that is created after clicking on **Learn Files** in the Batch Class Management administrative interface. The HOOCR file is the XML representation of the OCR output.

The XML file can be edited to remove any content that is not part of the blank form. After the XML file is updated, click on **Learn Files** again to update the index files used by Ephesoft. This will not overwrite the changes that have been made to the XML file; this will only happen if the source TIFF is updated.

Setting up Active Directory

Out of the box, Ephesoft uses Tomcat for authentication, authorization, and group management. However, in most production environments, this does not suffice and other external services, such as Active Directory, can be configured to perform these functions. Active Directory is Microsoft's implementation of a directory service based on the industry standard LDAP protocol.

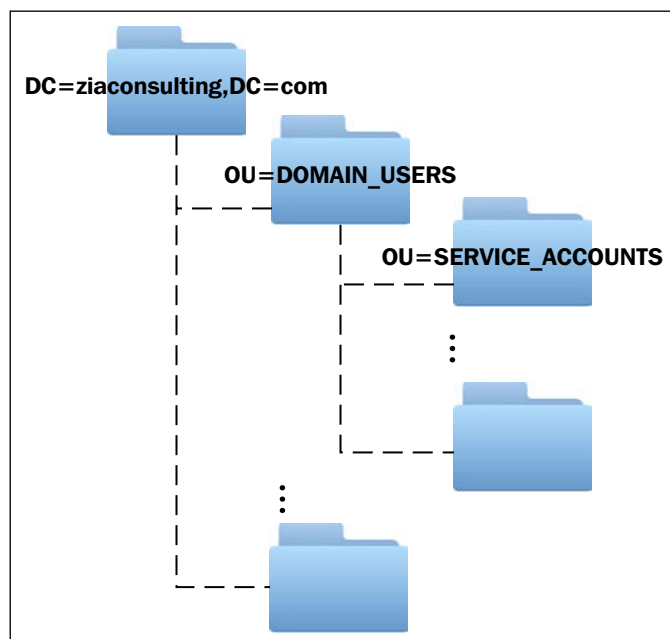
An Overview of LDAP

LDAP, or **Lightweight Directory Access Protocol**, is a standards-based protocol for communicating with a directory server. The structure of the protocol itself is not important, because the details of this protocol are hidden behind an API in all practical applications of the protocol. However, you should understand that it can be used to authenticate users, search directory paths, and view object attributes. Directory servers monitor configured TCP ports on host servers for inbound LDAP requests. SSL may be employed within the transport layer in order to ensure encrypted communication between the server and the client. An SSL-enabled LDAP connection is referred to as an LDAPS connection.

The primary function of a directory server application is to store information about users and groups within an enterprise. This includes information about who these users are, what organizations they belong to, what groups they belong to, and what their login credentials are for a related domain. The idea is that rather than maintaining user information and credentials across many different systems and software applications, an IT enterprise includes a single directory server in order to centralize this information. Application systems defer to this server in order to authenticate users and obtain information about their group membership so that authorization may be granted within each context.

Information is organized in a hierarchical fashion within a directory. At the root level of the directory structure is a domain component, or DC, which is similar in concept to a disk volume name. The DC name conforms to the domain name that is applied to the server host domain. In accordance with LDAP conventions, the . character is replaced by a comma and DC= is added as a prefix to each part of the name. Therefore, `ziaconsulting.com` becomes `DC=ziaconsulting,DC=com`.

Folders within the directory structure are called **organization units**, or **OUs**. They are containers for other OUs or user and group objects. Like folders, they are simply used in order to organize user and group objects so that they can be easily found and uniquely distinguished. Instead of using slashes as path separators, commas are used and OU= is prefixed to each OU in the path. Another important thing to note is that directory paths are constructed in the order of the most specific to the least specific, which is the reverse of how one typically constructs file system folder paths. For example, the directory path "`OU=SERVICE_ACCOUNTS,OU=DOMAIN_USERS,DC=ziaconsulting,DC=com`" is visually represented as follows:

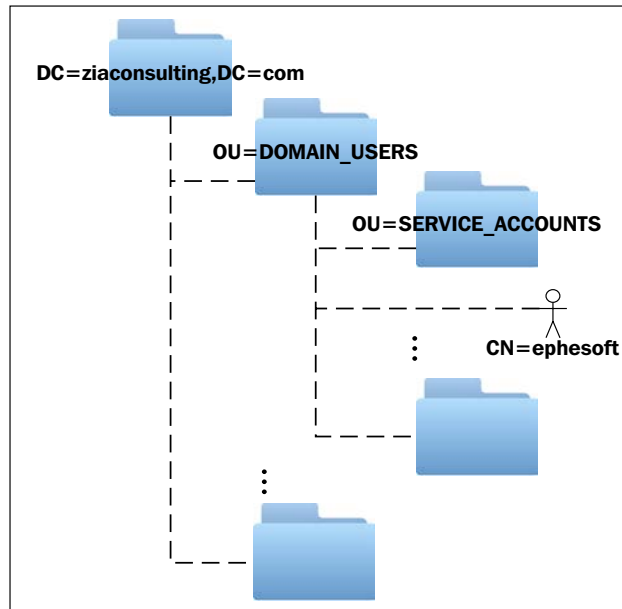


A sample LDAP structure

User and group objects within the directory structure have attributes associated with them that describe their characteristics. User attributes capture information such as the user's login ID, first name, last name, reporting manager, and location. New attributes can be created on the fly in order to capture any information that should be shared with the applications that require this information. There are several core attributes that are defined for user objects:

- **CN (Common Name)**: This attribute is typically used to capture a user's first name. It can be used for many other purposes, but it is used for first names within the enterprise.
- **SN (Surname)**: This attribute is typically used to capture a user's last name.
- **UID or sAMAccountName (User Login ID)**: Either one of these attributes may be used to capture a user's login ID within the domain. *sAMAccountName* is typically used within Microsoft Active Directory implementations of Directory Server.
- **DN (Distinguished Name)**: The DN is the unique name for the user within the directory. It represents the user object's unique path within the directory structure.

For example, CN=ephesoft,OU=SERVICE_ACCOUNTS,OU=DOMAIN_USERS,DC=ziaconsulting,DC=com uniquely identifies the Ephesoft user object within the directory for the DC= ziaconsulting,DC=com domain.



A sample LDAP structure with a user

Group objects share many of the same attributes, although some attributes such as SN do not apply. The member attribute is unique to the group object. It may be repeated within a single group object as a way of persisting the DNs of all user objects who are members of the group.

Ephesoft directory server service account

In order to be able to authenticate users and/or determine whether users are members of directory server groups, it is necessary for Ephesoft to connect to Active Directory by using a service account. This service account is identified by a DN and a password. Therefore, be sure to obtain the DN and the password for this service account prior to an Ephesoft Active Directory configuration attempt. The service account in the preceding example is: CN=ephesoft,OU=SERVICE_ACCOUNTS,OU=DOMAIN_USERS,DC=ziaconsulting,DC=com

Ephesoft directory server groups

Ephesoft must be configured to locate groups within a directory so that it might examine the member attribute values for each group and determine whether an authenticated user is a member of these groups. Group membership is required in order for users to gain access to batches that are ready for verification in any one of the configured batch classes, or to perform administrative activities within the administrative interfaces of the application.



It is a requirement of the Ephesoft Active Directory implementation that the DNs associated with directory user objects that are to be considered members of the group be added directly as new member attribute values to a related group. It is not sufficient to add the DN of another group as a member attribute of the Ephesoft group in order for users who are members of that group to be considered members of the Ephesoft group. The concept of nested groups is not recognized within the Ephesoft Active Directory implementation.

Ephesoft Active Directory configuration files

There are four configuration files associated with the Ephesoft Active Directory configuration that must be updated in order for Ephesoft to reference an external directory server. Once these files are updated, none of the user accounts are defined in the Tomcat users' configuration that is used for authentication and permissions when initially installed.

- `JavaAppServer\conf\server.xml`: This configuration file not only defines the Ephesoft application context within the embedded Tomcat application server, but also defines the Active Directory security realm for the application. Ephesoft leverages Tomcat's built-in security realm capability in order to authenticate users and obtain information about their directory server group memberships.
- `Application\WEB-INF\classes\META-INF\dcma-user-connectivity\user-connectivity.xml`: This file defines the Active Directory configuration for the Ephesoft Active Directory connector so that it is able to find Ephesoft related groups in Active Directory. The list of groups found in Active Directory using these settings is used to populate the roles list within the Ephesoft Batch Class Management interface. The roles selected in this list define the groups of users who have access to the related batch class.

- Application\WEB-INF\classes\META-INF\application.properties: This file has the configuration for the super admin group. Members of the super admin group are given permissions to access as batch classes and batch instances.
- Application\WEB-INF\web.xml: This configuration file not only defines the Java web application context for the entire Ephesoft application but also defines role-based security constraints on specific Ephesoft interfaces.

The following sections define how these configuration files are edited in order to communicate with a Microsoft Active Directory server.

server.xml

The following XML illustrates sample settings for the dcma.xml configuration file:

```
<Context path="/dcma" docBase="E:\Ephesoft\Application" debug="10"
privileged="false">
  <Realm className="org.apache.catalina.realm.JNDIRealm"
debug="99"
  connectionURL="ldap://elm.ziaconsulting.com:3268"
  connectionName="CN=ephesoft,OU=SERVICE_ACCOUNTS,
    OU=DOMAIN_USERS,DC=ziaconsulting,DC=com"
  connectionPassword="secret"
  referrals="follow"
  userBase="OU=DOMAIN_USERS,DC=ziaconsulting,DC=com "
  userSearch="(sAMAccountName={0}) "
  userSubtree="true"
roleBase="OU=DOMAIN_GROUPS,DC=ziaconsulting,DC=com"
  roleName="cn"
  roleSearch="(member={0}) "
  roleSubtree="true"/>
</Context>
```

Only the attributes of the Realm element are to be modified in this file. Always back up the file prior to making modifications. If the backup file is retained in the C:\Ephesoft\JavaAppServer\conf\Catalina\localhost folder along with the original, then be sure to change the file extension to something other than .xml.

The following table defines the purpose of each configuration attribute of the Realm element:

Attribute	Description
connectionURL	This specifies the protocol and URL for connecting to the Microsoft Active Directory server. <code>ldap</code> indicates that the protocol is LDAP. If LDAP over SSL is employed, then change the protocol designation from <code>ldap</code> to <code>ldaps</code> . The related SSL certificate must be installed in the Java-trusted keystore in order for the "ldaps" protocol to be used.
connectionName	This specifies the DN of the user object in the directory that is associated with the service account used by Ephesoft to connect to the directory server.
connectionPassword	This specifies the password for the user object in the directory that is associated with the service account used by Ephesoft to connect to the directory server.
referrals	This specifies whether the driver should search from the domain root of Active Directory for users. This attribute must be set to a value of <code>follows</code> when the target directory server is Microsoft Active Directory (as in this example).
userBase	This specifies the DN of the base directory where the user search should begin. We would like Ephesoft to authenticate all domain users, so the value of this attribute is set to <code>OU=DOMAIN_USERS,DC=ziaconsulting,DC=com</code> . Only user accounts that reside in this OU or a sub-OU will be visible.
userSearch	This specifies the attribute filter that should be used to find the user object within the directory. The <code>{0}</code> text is substituted with the user-entered login ID during the authentication process. Therefore, the fundamental purpose of this configuration value is to identify the attribute name of the user object in the Microsoft Active Directory server that should be matched to the user-entered text for the login ID.
userSubtree	This indicates that the search for the user object should include any sub-directory structure that may exist below the specified <code>userBase</code> DN.
roleBase	This specifies the base DN in the directory server where the Ephesoft roles can be found.

Attribute	Description
roleName	This specifies the group object attribute that defines the group name. Within the Microsoft Active Directory server, it is typical for both the sAMAccountName and the cn attributes to be used for the group name.
roleSearch	This specifies the attribute filter that should be used within a group object context in order to determine whether a user is a member of the group. The member attribute of a group object is typically used within the Microsoft Active Directory server for this purpose. It is this attribute value that is set to the DN of a user object in order to indicate group membership, so we set this filter value to member={0}. {0} is substituted with the user object DN.
roleSubtree	This indicates that group objects within subdirectories of the configured roleBase should be examined for user membership.

Once this configuration file has been updated, it is necessary to restart the Tomcat server that hosts the Ephesoft application in order for the changes to take effect. If the configuration is incorrect, then it will not be possible to log in to Ephesoft by using valid Active Directory credentials.

user-connectivity.xml

The following text illustrates sample settings for the user-connectivity.xml configuration file:

```
# This property is defined common for all types connectivity
# LDAP/MS Active Directory
user.connectivity_url=ldap://localhost:389
user.connectivity_config=com.sun.jndi.ldap.LdapCtxFactory
user.connectivity_domain_component_name=ziaconsulting
user.connectivity_domain_component_organization=com
user.connectivity_username=CN=ephesoft,OU=SERVICE_ACCOUNTS,OU=DOMA
IN_USERS,DC=ziaconsulting,DC=com
user.connectivity_password=secret
# This Property defines which type of connectivity is used
# 0 = LDAP
# 1 = MS Active Directory
# 2 = Tomcat
user.connection=1
# Set this for LDAP Connectivity
user.ldap_user_base=ou=people
user.ldap_group_base=ou=groups
```

```

#This Attribute is added so as to make search of groups in LDAP/AD
configurable, by default its cn(commonName) is returned
user.connectivity_groupSearchAttributeFilter=cn
#This Attribute is added to make search of Users (Organizational
Unit) in LDAP/AD configurable, by default its cn
user.connectivity_userSearchAttributeFilter=cn
#Set this for MS Active Directory
user.msactivedirectory_context_path=ou=DOMAIN_GROUPS
# filter can have |(OR), &(AND) and !(NOT)
# | (|(cn=a*))
# & (&(cn=a*))
# ! (!(cn=a*))
# complex example ((!(cn=a*))(|(cn=ephesoft*)&(cn=b*)))
user.msactivedirectory_group_search_filter=
# Tomcat Connectivity
user.tomcatUserXmlPath=C:\\Ephesoft\\JavaAppServer/conf/tomcat-
users.xml
#Switch To display user's Full name on the application UI.
# Default value is OFF.
# 1 = ON.
fullname.display=1

```

The properties that begin with `user.ldap` and `user.tomcat` may be ignored in this configuration file, as they pertain to non-Active Directory settings. Always back up the `user-connectivity.xml` file before making changes to the file. If the backup file is retained in the `C:\\Ephesoft\\Application\\WEB-INF\\classes\\META-INF\\dcma-user-connectivity\\user-connectivity.xml` folder along with the original, then be sure to change the file extension to something other than `.xml`.

The following table defines the purpose of each pertinent configuration property:

Attribute	Description
<code>user.connectivity_url</code>	This specifies the protocol and URL for connecting to the Microsoft Active Directory server. <code>ldap</code> indicates that the protocol is LDAP. If LDAP over SSL is employed, then change the protocol designation from <code>ldap</code> to <code>ldaps</code> . The related SSL certificate must be installed in the Java-trusted keystore in order for the <code>ldaps</code> protocol to be used.
<code>user.connectivity_config</code>	This specifies the name of the LDAP context implementation class to use. In this case, the built-in Oracle Java class is being referenced.

Attribute	Description
<code>user.msactivedirectory_context_path</code>	This specifies the base DN, minus the DC designations, in the directory server where the Ephesoft roles may be found. Multiple paths may be specified. Use ; ; as a delimiter between each of the specified paths.
<code>user.connectivity_domain_component_name</code>	This specifies the DC below the root DC, or <code>ziaconsulting</code> in our case, since the full domain is <code>DC=ziaconsulting,DC=com</code> .
<code>user.connectivity_domain_component_organization</code>	This specifies the root DC, or <code>com</code> in our case, since the full domain is <code>DC=ziaconsulting,DC=com</code> .
<code>user.connectivity_username</code>	This specifies the DN of the directory server account that should be used for querying the directory. In this example, the Microsoft Active Directory service account user is <code>ephesoft</code> , which has a DN of <code>CN=ephesoft,OU=SERVICE_ACCOUNTS,OU=DOMAIN_USERS,DC=ziaconsulting,DC=com</code> .
<code>user.connectivity_password</code>	This specifies the password associated with the service account specified in the previous property.
<code>user.msactivedirectory_group_search_filter</code>	This specifies a filter that may be used to filter the names of groups that may appear in the Ephesoft directory server group list—the roles list in the Batch Class Management interface. In this example, all of the Ephesoft group names begin with <code>Ephesoft</code> , and the <code>cn</code> attribute of the group object specifies the group name in the Microsoft Active Directory. Therefore, the filter <code>((cn=Ephesoft*))</code> may be specified to ensure that only group names that start with <code>Ephesoft</code> are included in the list.
<code>user.connection</code>	This specifies which type of source Ephesoft should use for obtaining the list of groups. A value of 1 is specified as the value of this property so that the <code>user.msactivedirectory</code> properties are utilized.

Once this configuration file has been updated, it is necessary to restart the Tomcat server that hosts the Ephesoft application in order for the changes to take effect.

To verify that the configuration is correct:

1. Navigate to the Batch Class Management interface for the deployment at `http://%hostname%:%port%/dcma/BatchClassManagement.html`
2. Log in using valid directory server credentials.
3. Select one of the batch classes, and click on the **Edit** button.
4. Click on the **Edit** button that appears in the **Batch Class Configuration** panel. The role list box will display each of the Ephesoft groups.
5. Highlight one or more roles in this list to ensure that users associated with this directory server group have access to the related batch class.

application.properties

The configuration file has two properties that determine the super admin group. The following is an example of how to reset the super admin group to a group named Ephesoft Admins:

```
user.super_admin=Ephesoft Admins
update_super_admin_group=true
```

Property	Description
<code>user.super_admin</code>	Members of this group will be given access to all batch classes and batch instances.
<code>update_super_admin_group</code>	This property should be set to true when the <code>super_admin</code> property is updated. This will modify the database to allow complete access to any existing batch classes and any new batch classes created.

web.xml

This configuration file not only defines the Java web application context for the entire Ephesoft application, but also defines role-based security constraints for specific Ephesoft interfaces. Each security constraint is defined in a `security_constraint` configuration block similar to the following:

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>batch class management</web-resource-name>
    <url-pattern>/BatchClassManagement.html</url-pattern>
    <http-method>GET</http-method>
    <http-method>POST</http-method>
  </web-resource-collection>
```

```
<auth-constraint>
  <role-name>Ephesoft Admins</role-name>
</auth-constraint>
</security-constraint>
```

It is the `role-name` element that specifies the directory server group to which the related interface is restricted. Therefore, in the previous example, the Batch Class Management interface is restricted to members of the `Ephesoft Admins` group. This same role name must be specified for the following security constraint web resources (identified by the `web-resource-name` element):

- batch class management
- batch instance management
- web scanner
- reporting
- upload batch

After making updates to the `web.xml` file, you must restart the Ephesoft service in order for these changes to take effect.

Active Directory troubleshooting

If you find that you are having problems verifying the Active Directory connector settings in either of these files, then download and install a free directory server management application such as Apache Directory Studio, and use the application to debug your settings. Directory Studio includes a search capability that functions using filters similar to those that were populated in the XML files.

Setting up e-mail processing

Ephesoft has the ability to poll POP3 and IMAP accounts. If Ephesoft finds new mail, it will create a new batch. The body of the e-mail will be converted to an image and included in the batch along with any attachments. Ephesoft can even process Microsoft Office files, such as Word documents.

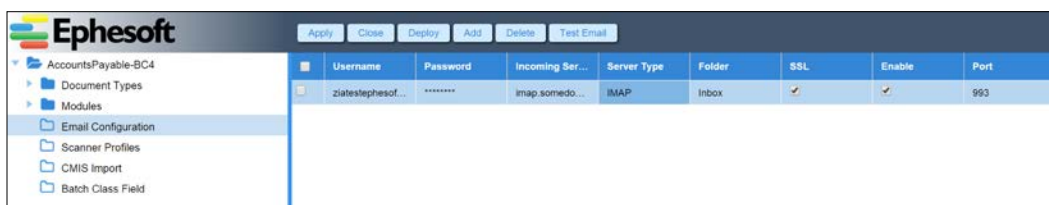


Ephesoft uses LibreOffice to convert documents of various formats. These formats can also be supported by editing the folder-monitor properties and adding to the valid extensions.



Run `netstat -a` from the command line to make sure the LibreOffice process starts correctly. It should be listening on port 8100.

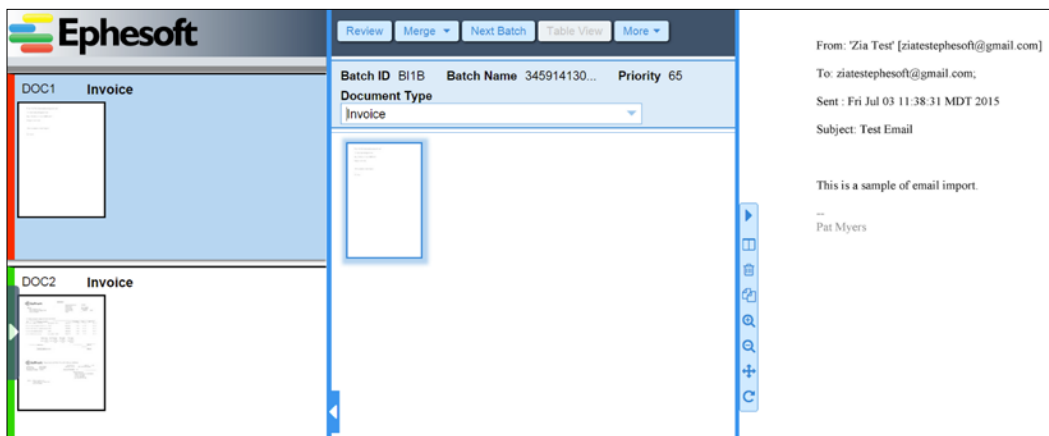
Once this is done, edit the batch class and go to the **Email Configuration** selection in the desired batch class. Click on the **Add** button and enter the authentication information for the e-mail account:



E-mail import configuration

Once the properties are entered, you can click on the **Test Email** button to test the connection. Now, send an e-mail to the account specified in **Email Configuration**; Ephesoft should create a new batch instance to process the e-mail.

By default, the e-mail body will be included in the e-mail batch:



Reviewing an imported e-mail



E-mail processing can be modified using the mail import properties. E-mail bodies can be dropped, more extensions of attachments can be supported, or multiple e-mails can be included in a single batch.

Summary

This concludes our investigation of document capture with Ephesoft. We started off with a review of document capture and followed it with a tour of the Ephesoft user interface, then walked you through the creation of a batch class and the processing of a batch instance. We then investigated increasingly complicated document capture problems, concluding with some tips from experienced Ephesoft implementation professionals.

There's always more to learn, and new features being added. You should visit Ephesoft's website and wiki regularly to stay informed. We thank you for your interest, and wish you luck in your implementation.

References

In this chapter, we provide definitions to help you become more familiar with the capture system terminology, as well as the tools that are specific to Ephesoft. The chapter includes the following:

- Glossary
- Common regular expressions
- Commonly-modified DCMA settings

Glossary

The following terms are commonly used when implementing document capture:

- **Batch class:** A definition of document types, associated fields, extraction rules, monitored folders, and e-mails for a specified workflow
- **Batch instance:** The pages being processed in the workflow
- **Classification :** Determining the type of document being processed
- **CMIS:** Content Management Interoperability Services
- **CMS:** Content Management System
- **DMS:** Document Management System, another term for CMS
- **ECM:** Enterprise Content Management, an enterprise application for managing a large number of documents
- **Extraction:** Retrieving information from documents
- **Fixed form:** A type of form where the positions and dimensions of the fields are always the same

- **HA:** High Availability, a term applied to online applications, services, or technologies that are designed to be resistant to failure, and therefore, always accessible
- **Hand print:** Hand-written text
- **ICR:** Intelligent Character Recognition
- **IDC:** Intelligent Document Capture
- **Indexing:** The process of defining field values for a particular document instance
- **KV:** A key-value pair
- **Lucene:** A full-text search engine
- **Machine print:** Text that is printed by a machine (not hand-written)
- **Metadata:** Information about a document that is associated with that document but not stored in the body of the document itself
- **OCR:** Optical Character Recognition
- **OOTB :** Out-of-the-box, refers to the default configuration of an application
- **Regex:** A regular expression, syntax for defining a pattern of text
- **Separation:** The process of determining the start and end of documents, given a set of page images
- **UI:** User Interface
- **WSDL:** Web Service Definition Language

Common regular expressions

The regular expressions used in Ephesoft are Java regular expressions. The reference documentation can be found on the Oracle website.

The following describes some more commonly used patterns:

- **Date:** `[0-9]{1,2}/[0-9]{1,2}/[0-9]{2,4}`

This pattern will look for 1 or 2 digits, `[0-9]{1,2}`, followed by a `/` and then 1 or 2 digits, `[0-9]{1,2}` followed by a `/` followed by 2 or 4 digits, `[0-9]{2,4}`. Examples of matching patterns are `1/31/12` and `03/17/1974`.

- **Currency:** `[0-9]{1,3}?,?[0-9]{1,3}\.[0-9]{2}`
This pattern will look for 1 to 3 digits, `[0-9]{1,3}`, followed by a `,` and then, by 1 to 3 digits followed by a `"` that is followed by 2 digits. The `?` means a 0 or 1 instance of the pattern, so in this case, anything followed with the `?` is optional. Examples of this pattern are `20.00`, `50000.00`, and `600,000.00`.
- **Name with Letters Only:** `[a-zA-Z]{2,25}`
This pattern will look for any text that contains only 2 to 25 upper and lower case alpha characters.



The following characters need to be escaped with a `"\ " - "[\ ^$.|?*(+){}."`

Commonly-modified Ephesoft settings

Ephesoft has a lot of configuration can be modified to optimized your implementation. Let's review some of the more common settings.

dcma-workflow.properties

The number of workflow threads can be specified here. This can be changed depending on the number of cores on the Ephesoft server. If the architecture only has a single server and CPU, this number may need to be lowered to have more processor availability for operators using the review and validation screens.

```
server.instance.max.process.capacity=4
```

dcma-batch.properties

This file contains the configuration for batch processing:

- `batch.local_folder`: The location where in-process batch information and transformations are kept, for example: `batch.local_folder=C:\\Ephesoft\\SharedFolders\\ephesoft-system-folder`.
- `batch.base_http_url` base url: Used for the location of thumbnails and preview images for the application. The host should be modified to the web host server name or proxy server configuration, for example: `batch.base_http_url=http://ServerA.acme.com:8080/dcma-batches`.

dcma-cmis.properties

This file has the settings for CMIS export:

- `cmis.date_format`: The Java date format used to transport date type fields to date attributes in CMIS. This is a global setting for all the batch classes on the server. Dates must be normalized before the CMIS export, for example: `cmis.date_format=MM/dd/yyyy`.
- `cmis.document_versioning_state`: This specifies the versioning strategy for items exported from Ephesoft. Options are available as comments in the properties file, for example: `cmis.document_versioning_state=NONE`.

Index

A

Active Directory

- Ephesoft Active Directory, configuration files 121
- LDAP 118
- setting up 117
- troubleshooting 128

Active Directory configuration files,

Ephesoft

- about 121
- application.properties 122, 127
- server.xml 121-123
- user-connectivity.xml 121-127
- web.xml 122-128

administrative features

- areas 2
- batch class management 2-4
- batch instance management 4
- folder management 5
- operator user interface 7
- reports 7
- system configuration 6

application script 98

attributes, Realm element

- connectionName 123
- connectionPassword 123
- connectionURL 123
- referrals 123
- roleBase 123
- roleName 124
- roleSearch 124
- roleSubtree 124
- userBase 123

userSearch 123

userSubtree 123

attributes, user objects

- Common Name (CN) 119
- Distinguished Name (DN) 119
- sAMAccountName (User Login ID) 119
- Surname (SN) 119

automatic classification 42

B

barcode classification

- about 42, 72, 73
- Barcode Reader plugin, enabling 72
- RecoStar HOCR plugin, enabling 72

BareTail

- URL 116

batch

- restarting 117
- uploading 57, 58

batch class

- about 13, 131
- classification 18-20
- configuration management 66-69
- creating 14, 15
- document, reviewing 38
- document validation 37-40
- exporting 32-34
- file upload 35, 36
- processing 34
- separation 18-20
- starting, from other sources 37

batch class management interface

- batch class fields 4
- CMIS import 3

- document types 3
- e-mail configuration 3
- modules 3
- scanner profiles 3
- batch instance** 13, 131
- blank forms**
 - non-availability, for training 117

C

- classification methods**
 - barcode classification 72, 73
 - configuring 71
 - image classification 73
- CMIS**
 - about 59, 131
 - content model, establishing 59
 - document type 61
 - export plugin, configuring 60, 61
 - global configuration 62
 - property mapping 61
- commonly-modified Ephesoft settings**
 - about 133
 - dcma-batch.properties 133
 - dcma-cmis.properties 134
 - dcma-workflow.properties 133
- confidence scores, calculating**
 - about 42
 - for automatic classification 47
 - for barcode classification 46
 - for image classification 47
 - for search classification 43-46
- configuration property**
 - attributes 125, 126
- Content Management Interoperability Services.** *See* CMIS
- Copy Batch XML** 32-34
- custom plugin**
 - adding, to Ephesoft 105, 106
 - writing 100-105
- custom workflow**
 - about 99
 - batch class workflows, customizing 99
- Cygwin**
 - URL 116

D

- DMS** 131
- document type**
 - creating 16-18

E

- ECM** 131
- e-mail processing**
 - setting up 128, 129
- Ephesoft**
 - Active Directory configuration files 121
 - administrative features 2
 - custom plugin, adding 105, 106
 - file system 11
 - search classification 41
 - troubleshooting 115
 - user interface 1
- export**
 - about 58
 - CMIS export 59
 - database export 63, 64
 - plugins 66
- extraction** 25, 131

F

- fields**
 - creating 21-24
- fixed form extraction**
 - Ephesoft configuration, for using RecoStar project 81, 82
 - RecoStar project, configuring 79, 81
 - RecoStar project, creating 75-78
 - using 73, 74
- fuzzy database (Fuzzy DB)**
 - about 50-53
 - testing 54

H

- HA** 132
- hand print** 132

I

ICR 132

IDC 132

image classification 42, 73

invoice processing portal

about 108

controller, creating 110, 111

upload view, creating 109

using 112

web services, calling 110, 111

K

key/value extraction 25-30

KV 132

L

Lightweight Directory Access Protocol (LDAP)

Ephesoft directory server groups 121

Ephesoft Directory Server service

account 120

overview 118-120

Lucene 132

Lucene classification. *See* search classification

M

machine print 132

metadata 132

O

OCR 132

OOTB 132

operator user interface

about 7

batch list 8

review validate screen 9

upload batch feature 10

web scanner 10

organization units (OUs) 118

R

regular expressions

about 132

currency 133

date 132

Name with Letters Only 133

S

scripts

about 88

application script 98

triggering, from Function key 95-97

triggering, on field edit 93, 95

workflow scripts 88-92

search classification

about 41-45

confidence scores 42

Document Assembler plugin,
responsibility 43

types 41

search classification, types

about 41

automatic 42

barcodes 42

image 42

one document classification 42

search 41

single document type

multiple layouts 47-50

Ephesoft installation folder

subfolders 11, 12

T

table extraction

using 82-88

troubleshooting

about 115

batch file, examining 116

batch progress, monitoring 116

logging 115, 116

U

UI 132

user interface 1

V

validation rules 30

W

Web Scanner

using 55, 56

web services

about 107

Create Searchable PDF 107

Execute Mobile Upload 107

Extract KV for Document Type 107

Get Batch Instance List 107

OCR Classify Extract 107

Web Services API

about 106

enabling 107

example 108

web services 107

Web Service Security (WSS) 62

workflow scripts 88-92

WSDL 132



Thank you for buying
Intelligent Document Capture with Ephesoft
Second Edition

About Packt Publishing

Packt, pronounced 'packed', published its first book, *Mastering phpMyAdmin for Effective MySQL Management*, in April 2004, and subsequently continued to specialize in publishing highly focused books on specific technologies and solutions.

Our books and publications share the experiences of your fellow IT professionals in adapting and customizing today's systems, applications, and frameworks. Our solution-based books give you the knowledge and power to customize the software and technologies you're using to get the job done. Packt books are more specific and less general than the IT books you have seen in the past. Our unique business model allows us to bring you more focused information, giving you more of what you need to know, and less of what you don't.

Packt is a modern yet unique publishing company that focuses on producing quality, cutting-edge books for communities of developers, administrators, and newbies alike. For more information, please visit our website at www.packtpub.com.

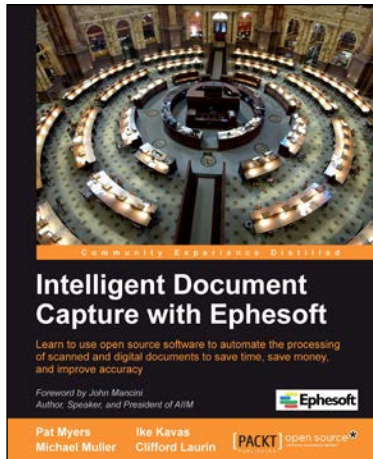
About Packt Enterprise

In 2010, Packt launched two new brands, Packt Enterprise and Packt Open Source, in order to continue its focus on specialization. This book is part of the Packt Enterprise brand, home to books published on enterprise software – software created by major vendors, including (but not limited to) IBM, Microsoft, and Oracle, often for use in other corporations. Its titles will offer information relevant to a range of users of this software, including administrators, developers, architects, and end users.

Writing for Packt

We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to author@packtpub.com. If your book idea is still at an early stage and you would like to discuss it first before writing a formal book proposal, then please contact us; one of our commissioning editors will get in touch with you.

We're not just looking for published authors; if you have strong technical skills but no writing experience, our experienced editors can help you develop a writing career, or simply get some additional reward for your expertise.



Intelligent Document Capture with Ephesoft

ISBN: 978-1-84969-372-1

Paperback: 182 pages

Learn to use open source software to automate the processing of scanned and digital documents to save time, save money, and improve accuracy

1. Learn the benefits of intelligent document capture and how to implement document capture using Ephesoft.
2. Capture relevant information from your documents, even if they vary widely in format and appearance.
3. Leverage the power of open source software to implement a cost effective solution for document capture.



Web Content Management with Documentum

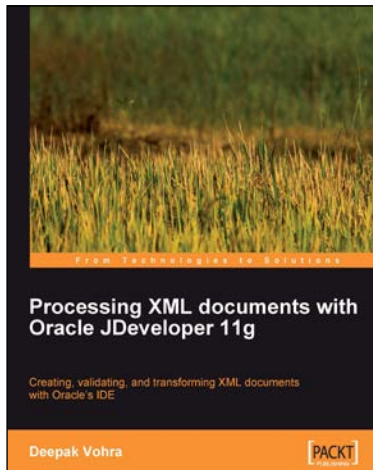
ISBN: 978-1-90481-109-1

Paperback: 484 pages

Concise, practical information on Documentum Web Content Management to get the most from this system

1. Design and implement Documentum applications.
2. Practical examples to help you get the most from Documentum.
3. Tips and tricks to ease everyday working with the system.

Please check www.PacktPub.com for information on our titles



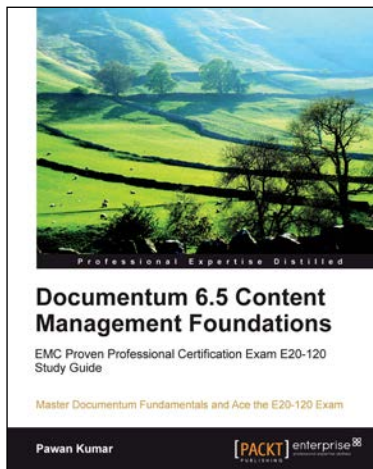
Processing XML documents with Oracle JDeveloper 11g

ISBN: 978-1-84719-666-8

Paperback: 384 pages

Create, validate, and transform XML documents with Oracle's JDeveloper IDE

1. Will get the reader developing applications for processing XML in JDeveloper 11g quickly and easily.
2. Self-contained chapters provide thorough, comprehensive instructions on how to use JDeveloper to create, validate, parse, transform, and compare XML documents.
3. The only title to cover XML processing in Oracle JDeveloper 11g, this book includes information on the Oracle XDK 11g APIs.



Documentum 6.5 Content Management Foundations

ISBN: 978-1-84968-022-6

Paperback: 416 pages

Master Documentum Fundamentals and Ace the E20-120 Exam

1. Technical foundations of the Documentum platform.
2. Over 200 practice questions and three practice tests.
3. Up-to-date information on version 6.5 SP2 including presets, aspects, new Webtop interface, lightweight types, Composer, and DFS.

Please check www.PacktPub.com for information on our titles