Redbooks
ibm.com/redbooks

Implementing Document Imaging and Capture Solutions with IBM Datacap

Whei-Jen Chen

Ben Antin

Kevin Bowe

Ben Davies

Jan den Hartog

Daniel Ouimet

Tom Stuart

ECM

IBM

Redbooks

International Technical Support Organization

Implementing Document Imaging and Capture Solutions with IBM Datacap

October 2015

SG24-7969-01

Note: Before using this information and the product it supports, read the information in "Notices" on page xi.

Second Edition (October 2015)

This edition applies to IBM Datacap Version 9.

Contents

Find and read thousands of IBM Redbooks publications

- Search, bookmark, save and organize favorites
- Get up-to-the-minute Redbooks news and announcements
- Link to the latest Redbooks blogs and videos

Get the latest version of the Redbooks Mobile App

iOS **Download Now** Android

Promote your business in an IBM Redbooks publication

Place a Sponsorship Promotion in an IBM® Redbooks® publication, featuring your business or solution with a link to your web site.

Qualified IBM Business Partners may place a full page promotion in the most popular Redbooks publications. Imagine the power of being seen by users who download millions of Redbooks publications each year!

ibm.com/Redbooks
About Redbooks → Business Partner Programs

THIS PAGE INTENTIONALLY LEFT BLANK

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

Daeja™
DB2®
developerWorks®
FileNet®
IBM®
Redbooks®
Redbooks (logo) ®
WebSphere®

The following terms are trademarks of other companies:

Inc., and Inc. device are trademarks or registered trademarks of Kenexa, an IBM Company.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

Preface

Organizations face many challenges in managing ever-increasing documents that they need to conduct their businesses. IBM® content management and imaging solutions can capture, store, manage, integrate, and deliver various forms of content throughout an enterprise. These tools can help reduce costs associated with content management and help organizations deliver improved customer service. The advanced document capture capabilities are provided through IBM Datacap software.

This IBM Redbooks® publication focuses on Datacap components, system architecture, functions, and capabilities. It explains how Datacap works, how to design a document image capture solution, and how to implement the solution using Datacap Developer Tools, such as Datacap FastDoc (Admin). FastDoc is the development tool that designers use to create rules and rule sets, configure a document hierarchy and task profiles, and set up a verification panel for image verification.

A loan application example explains the advanced technologies of IBM Datacap Version 9. This scenario shows how to develop a versatile capture solution that is able to handle both structured and unstructured documents. Information about high availability, scalability, performance, backup and recovery options, preferable practices, and suggestions for designing and implementing an imaging solution is also included.

This book is intended for IT architects and professionals who are responsible for creating, improving, designing, and implementing document imaging solutions for their organizations.

Authors

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center, in California, USA.

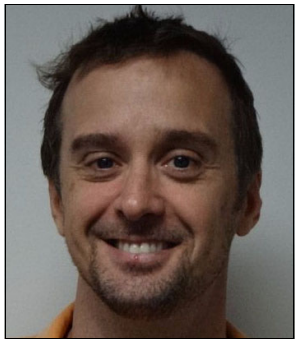
Whei-Jen Chen is a Project Leader for the International Technical Support Organization, San Jose Center. She has extensive experience in application development, database design and modeling, and IBM DB2® system administration. Whei-Jen is an IBM Certified Solutions Expert in database administration and application development and an IBM Certified IT Specialist.

Ben Antin is the IBM product manager for IBM capture and document imaging products. His has worked with Datacap products for four years and with IBM FileNet® enterprise content management (ECM) products for more than 20 years. Ben specializes applying automating and optimizing document processes using ECM technology. He has applied ECM technology in the financial services, insurance, transportation, and government sectors. He holds a degree in Computer Science from the University of Iowa.

Kevin Bowe is an IBM Client Technical Specialist who has specialized in enterprise content management (ECM) for more than 20 years. His experience includes business process re-engineering, system architecture, project management, release and regression testing, development, proposal creation and response, presentations, demonstrations, and hands-on experience with advanced capture and capture products. The majority of Kevin's ECM experience is derived from the IBM FileNet product suite.

Ben Davies works on the IBM worldwide team as a Smarter Content Solutions Leader, specializing in imaging and capture. He joined IBM in 2007 and has worked with IBM Datacap since the acquisition of Datacap Inc.® in 2010. He has built up a wealth of knowledge on the IBM Datacap product through assisting customers and partners around the world on a variety of engagements. Ben has been involved developing ideas and prototypes for new methods of image capture and data extraction. He was a co-author of the First Edition of *Implementing Imaging Solutions with IBM Production Imaging Edition and IBM Datacap Taskmaster Capture*, SG24-7969 and co-author of *Building IBM Enterprise Content Management Solutions From End to End*, SG24-8226. Ben holds a bachelor degree in Multimedia Computing from Liverpool John Moores University in the UK.

Jan den Hartog is an Enterprise Content Management (ECM) Technical Sales Specialist in the United States. Jan has more than 18 years of experience in product management, technical pre-sales, system integration, single-source publishing, and document transformation and migration. His experience, together with time spent in a support role, has taught him how to explain complex technical concepts in a clear and understandable manner.



Daniel Ouimet is a Technical Lead for IBM Datacap and Imaging for North America. Dan has over 18 years experience as lab services delivery consultant, delivery manager, technical architect, and presales technical resource for Enterprise Content Management in both US and Canada. Daniel possesses in-depth technical expertise in imaging, compliance, process automation, and content federation solutions. Daniel has published many technical papers and is the co-author of IBM Redbooks publication titled *Federated Content Management: Accessing Content from Disparate Repositories with IBM Content Federation Services and IBM Content Integrator*, SG24-7742.

Tom Stuart is the IBM Executive ECM capture alchemist and is part of the IBM North America Software Advanced Technology (SWAT) team. He is responsible for developing easy-to-replicate techniques that help customers rapidly deploy complex data capture applications, that are likely integrated with other business applications. Tom is also closely involved with refining prototype solutions and has the application developer responsible for IBM Datacap Taskmaster Accounts Payable Capture and the IBM Datacap product Taskmaster Flex. He previously directed the pre-sales engineering activities of Datacap. Tom has a dual background in education and product development. He is widely quoted on technical issues and frequently addresses industry conferences.

Acknowledgments

The authors thank Patrick Chesnot for his contribution in content and consultation for this book. Patrick is an Enterprise Content Management Product Manager with IBM Software Group US. He has more than 11 years of experience in FileNet ECM product management. He also has more than 22 years in the field of document management, imaging, and publishing systems, where he worked as a product specialist, systems consultant, and project manager.

Thanks to the following people for their contributions to this project:

Feri Clayton, Enterprise Content Management Director, document imaging, and capture solutions software developer

Noel Kropf, Enterprise Content Management Software Architect and IBM Datacap Architect

Nirmal K. Mukhi, Solutions Portfolio Manager, IBM Datacap Development

Charles F. Wiecha, Watson Conversational Dialog Management, IBM Software Group

Wei-Dong Zhu, Enterprise Content Management Solution Consultant, Enterprise Content Management Information Developer

Thanks to the authors of the First Edition of this book, which was published in October, 2011:

Jackie Zhu
Ben Antin
Moe Bryan
Patrick Chesnot
Ben Davies
Tom Stuart
Michael Vahland

Now you can become a published author, too

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time. Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form:

  ibm.com/redbooks

- Send your comments by email:

  redbooks@us.ibm.com

- Mail your comments:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Summary of changes

This section describes the technical changes made in this edition of the book and in previous editions. This edition might also include minor corrections and editorial changes that are not identified.

Summary of changes for
Implementing Imaging Solutions with IBM Production Imaging Edition and IBM Datacap Taskmaster Capture, SG24-7969
as created or updated on October 26, 2015

October 2015, Second Edition

This revision reflects the addition, deletion, or modification of new and changed information as described here.

New information

- IBM Datacap Version 9 new features and functions
- Solution examples

Changed information

- New branding name of IBM Datacap

1

Advanced imaging

This chapter provides an overview of *advanced imaging*, which is a set of capabilities for managing the entire document lifecycle, including a document capture solution, case management capabilities, and an enterprise content management (ECM) repository.

This chapter includes the following topics:

- The business document problem
- Advanced imaging
- Datacap components
- The advanced imaging process
- Examples of applications

1.1 The business document problem

Organizations today face many challenges in efficiently managing the documents they need to conduct their business. They have the perennial paper problem and the ever-increasing electronic document problem. One challenge is controlling all of the types of media that are required to conduct businesses. Another challenge is ensuring continuity, consistency, and longevity across these media over the entire lifecycle of the business processes. The approach to solving these problems is an advanced imaging solution.

1.1.1 Paper everywhere

The advent of such innovations as email, the web, instant messaging, and social media has resulted in more efficient communication and a significant reduction in the need to print information. Despite these technological advances, many companies still rely heavily on paper to conduct a large part of their businesses.

Organizations use paper for several reasons:

- Historical. The organization must deal with existing forms and documents and is unable to mandate the conversion of these paper forms and documents into electronic formats. Sometimes, the number of existing forms and documents make it unrealistic to undertake such a conversion process.
- Legal. In some cases, legislation has not kept pace with new technology and still refers specifically to keeping paper records (which holds probative value). In other cases, new laws take into account electronic media, but have not yet been challenged in court, which means that the safe option is to keep using paper.
- Practical. The low-tech portability of paper makes it the best option to reach customers anywhere, irrespective of affordability, access to infrastructure, technical dependencies, or administrative boundaries. For example, many companies that otherwise rely on electronic means of communication for marketing purposes still send correspondence by post to confirm important transactions that require the attention and signature of the customer. Similarly, many customers still prefer to send their official correspondence through registered mail to be tracked by a third party, with the expectation that it will be deemed an official record.
- Technical. Paper is the ultimate “systems integration technology,” a variant of the previous reasons. It is common to hear that the best way to communicate with another department within the same company or administration is to print the information and send it out. This is especially true in organizations where record keeping of inbound and outbound communication is available for mail but not across incompatible business or ECM systems.

1.1.2 Business challenges posed by paper

Even though many businesses cannot do without paper altogether, dealing with paper continues to pose several challenges.

For example, paper is expensive to store for the long term and to preserve under optimal conditions for business, legal, disaster (flood and fire), security, and safety reasons. Consider the cost incurred in maintaining shelf space, physical filing systems, and cabinets. Such costs also include robotics, temperature, and hygrometric control; fire and flood protection; and, periodically, verifying that the contents have not deteriorated over time or under heavy use.

Paper is inefficient, time-consuming, inflexible, and expensive to manipulate in the course of conducting day-to-day business. For example, a home loan or credit card department might receive hundreds or thousands of applications from customers each day. These hardcopy documents must be logged in to their systems and information about these applications must be entered into electronic loan processing systems. Manually handling these applications is inefficient, time-consuming, and prone to human error.

As another example, for a particular document, you might have to go to the file cabinet, control who has clearance to access the document, track who has had physical access to it, and flag it as checked out. You might also have to make copies of the document to share with people working on the case or project and write comments on the document to pass on to the next person in the process.

In a call center, it is inconceivable to operate this way. Still, for many organizations, productivity and access speed were not deemed critical before. However, they now find themselves in a situation where they can no longer afford to operate in this way as competitive pressures and volumes increase.

Physical documents are subject to human error, so they are more easily lost, misfiled, or misclassified and, sometimes, never recovered. Their contents can be misread and entered with errors. Contents can also be discovered or used beyond their authorized purpose. The consequences can be expensive for a business.

Most organizations are concerned with compliance. They want to ensure that they preserve the correct documents and discard documents that are no longer needed for the business. They also want to ensure that they purge documents as required after a certain period, as mandated by law, as is the case in some European countries. Although records management systems have long been available to manage physical records, it is increasingly challenging to manage records efficiently as the volumes of records increase.

The information about physical documents is on paper, which does not lend itself to automation and verification. As a result, transcription errors spread, and original mistakes go undetected. Spelling variations and typographical errors in names and nomenclatures cannot be verified and corrected until late in the business process, at which point it is more expensive to fix. Similarly, the business process cannot be automatically driven by the data. As a result, dedicated and expensive labor is required to carry out the low-value activity of routing documents to the next person in the process.

1.1.3 Business challenges posed by electronic documents

In many cases, organizations now use electronic documents in their business processes. For some of these processes, especially those processes that include external cycles with customers and business partners, it is still difficult to implement a solution that relies on one type of electronic media from end to end.

For example, it is now common to exchange electronic documents in at least part of the processing of a mortgage loan, from application to closing. However, especially in the case of a brokerage company, the organization might be bound by legal requirements for transacting business in paper format. In many cases, the organization cannot impose too many constraints on its customers and the other parties involved, such as lenders and assessors. As a result, the organization must accommodate many situations. So it continues to accept the receipt of paper documents alongside electronic ones, which leads to the need to handle multiple types of media over the course of working on a mortgage loan.

The loan application goes through the following process:

- The initial loan application form on the web typically feeds into a business database. The form is used to generate an initial loan offer that is sent to a customer on paper or electronically as a PDF document.
- Preliminary contacts between the broker and the customer generate several email messages with attachments back and forth.
- Additional forms (disclosures, authorizations, estimates, and so on), typically in PDF, are also exchanged. Signed copies must be captured, which leads to printing and faxing or scanning of documents that were originally generated electronically and then sent by email.
- The final loan settlement statement is signed. Each page is initialled by the customer at the formal closing meeting with the escrow officer. Then each page is scanned in and stored as the main document of record.

As you can see, although this is faster than if it were conducted entirely through paper and postal service, this process can be inefficient. It causes discontinuities in the media (from electronic text to paper to back to electronic image) and communication (email messages or faxes sent back and forth) over the lifecycle of a given document. This challenge makes it difficult to ensure data consistency, control, and reconciliation with the business process and the transactions of the in-house line of business (LOB) systems.

These documents are often archived over the long term in a specific, final form. This form must ensure preservation of the formatting and contents across multiple generations of technologies, without needing to use the original application that created it.

1.2 Advanced imaging

The challenges described in the previous section align closely with the business objectives many organizations hope to achieve with enterprise content management (ECM) systems, including advanced imaging. Advanced imaging helps organizations manage the documents they need to conduct their operations and meet their business goals more efficiently. It is a comprehensive set of technologies that enables organizations to implement several efficiencies:

- Reduce or eliminate the external paper cycle as much as possible by limiting or removing the need to print, copy, store, and manipulate paper, and, ultimately, reduce cost.
- Capture customer input at the source in virtual forms (such as self-service portals and electronic documents) to limit errors, reduce delays and unnecessary steps, optimize processes, and reduce labor.
- Integrate document processing systems and repositories with LOB systems to reduce the need for paper, reduce errors, normalize data, and minimize labor.
- Digitize paper documents to feed into internal business processes as early as possible. This objective allows sharing and optimizes the processing of documents downstream, while meeting the scalability and deployment requirements that are typical of large organizations.
- Automate scanning, classification, separation, and data extraction of paper documents to minimize the added labor cost of scanning operations.
- Automate the import and conversion of electronic documents and email messages by using the same infrastructure as the one used for paper scanning, thus benefiting from the same data normalization and validation rules.

- Verify and normalize data against business rules and databases and reduce errors. These types of issues are always more expensive to fix after they spread downstream.
- Provide flexibility to distribute image-processing operations (from local, to departmental, to central) to use available resources or easily shift them (such as when dispatching resources in disaster areas for the insurance industry).
- Store, classify or file, and secure content to avoid losses, provide protection, enforce compliance, manage records, and enable sharing.
- Provide context intelligence to drive automation of the business process and reduce delays, streamlining the allocation of work.
- Automatically deliver data and documents to users in a context that is the most relevant to the business process, using business process management technology. In addition, balance work loads, track work items, and gather statistics to help manage and tune the business processes and increase efficiency.

There are many examples of practical goals that some IBM clients have attained by implementing advanced imaging solutions.

For example, a university was able to dramatically cut costs and improve productivity and customer satisfaction. It freed up much of the storage space that was previously occupied by paper archives and streamlined the flow of information in the organization. In making these changes, the university saved significant time for their staff, enabling them to focus more on their priorities. They also improved customer service by providing faster access to customers' correspondence.

In another example, a state tax department was able to improve its operations while providing jobs to residents of the state. It reduced the labor-intensive work that was needed in scan preparation and fixing data entry errors. It was able to double data entry productivity and achieve accurate reporting on operations and resource utilization. The department can now also access easily any tax return in the system. Of equal importance, the tax department can now use local labor and stimulate low income areas of the state by offering remote document processing positions.

As yet another example, a global logistics company reduced penalties and fines by processing shipping documents faster and meeting service level agreements. At the same time the company improved compliance with regulations and reduced overall processing costs and the number full-time employees.

As a final example, a healthcare insurance company achieved an average of 50% reduction in personnel required to process claims. The company also achieved faster turnaround times, a reduction in duplicate claims submitted, and increased accuracy in claim processing.

By implementing advanced imaging solutions, these organizations solved many of their business document problems.

1.2.1 Components of an advanced imaging solution

Historically, part of the challenge of moving beyond paper was creating a solution with all of the capabilities that you need to achieve the business outcomes that you want. Today, a solution can be configured that provides advanced imaging capabilities by using *tightly integrated offerings from a single vendor*.

In addition to IBM Datacap, which is IBM's advanced document imaging and capture software and the focus of this book, the following software might be used:

- IBM FileNet Content Manager

  ECM software that includes IBM Content Navigator, collaborative and mobile content experience across the ECM portfolio, and IBM Daeja™ ViewONE, a document and image viewer and editor

- IBM Case Foundation

  Platform for content-centric business process management (BPM)

- IBM Case Manager

  Advanced case management (ACM) offering

IBM Production Imaging Edition includes several of these products in a single, pre-integrated offering. Its add-on components address various imaging use cases, including integrations with business applications and customer information databases.

You can find more information about all of the IBM ECM offerings on the "Content management and imaging" web page:

http://ibm.co/1KcIuBT

1.3 Datacap components

IBM Datacap handles production-level digitization, data extraction, verification, indexing, and exporting of documents to back-end systems. It includes the following components:

- Server components to manage and serve the images that have been scanned or imported
- Thick, thin, mobile, and multifunction devices (MFDs) embedded as clients to handle user-attended tasks such as scanning, indexing, verification of metadata, and administration of users and security
- Rulerunner, which is a rules engine to execute unattended capture operations such as image cleanup, data extraction, lookup, redaction, and export of contents and metadata to back-end systems
- Configuration, reporting, and system monitoring tools

Table 1-1 lists the Datacap components and functions.

Table 1-1 Datacap components

Component name	Functions
Datacap server	Manages and serves documents and executes the Datacap workflow
Datacap Rulerunner	Executes processing rules on documents
Datacap database server	Hosts Datacap databases for administering and controlling the processes of Datacap applications
Datacap Fingerprint Service	Caches and serves fingerprints to Datacap applications
Datacap Web	Provides a web interface for user-attended tasks and administration. This interface is based Microsoft Internet Information Server (IIS) and ASPX technology.
Datacap Navigator	Provides a web interface for user-attended tasks and administration. This interface is based on IBM Content Navigator, HTML5 and Dojo technology running in a Java Platform, Enterprise Edition application server
Datacap Desktop	Provides a thick client interface for running user-attended tasks
Datacap Mobile	Provides imaging and capture capabilities on iOS and Android devices.

Component name	Functions
Datacap FastDoc	Combines an entry-level user interface with business user-oriented tools for quickly setting up and testing Datacap applications. In addition, this thick client can run in stand-alone mode.
Datacap Studio	Provides advanced functions to develop, assemble rulesets, and configure and tests applications.
Datacap Application Manager	Maintains a registry of applications.
Datacap Report Viewer	Reports on runtime activity.
Datacap Maintenance Manager	Monitors operations and automates recurring system maintenance tasks.
Datacap Application Copy Tool	Copies or updates application databases and configuration files in a single process between operating environments.
Datacap License Manager	Registers and tracks the licenses of the base product and add-on components.
Rulerunner Enterprise	Multithreading and enhanced Fingerprint services
Connector for Email and Electronic Documents	Imports email attachments from Exchange and Internet Message Access Protocol (IMAP) mail servers and converts electronic documents
Connector for EMC Documentum	Exports document images and data to the EMC Documentum repository
Connector for Microsoft SharePoint	Exports document images and data to Microsoft SharePoint
Connector for Rightfax	Imports fax images from an OpenText RightFax server
Datacap Accounts Payable	Solution to process invoices and similar documents
Datacap Medical Claims	Solution to process US healthcare claim and explanation of benefits forms

Note: Datacap includes all of these components on its distribution media. However, some components require additional licensing from IBM.

Also, a set of sample applications that can be cloned and used as a base to give you a head start in configuring your own application are also available from the Datacap Technical Mastery section in IBM developerWorks® Communities:

http://ibm.co/1MwxWxW

1.4 The advanced imaging process

To understand the nature of the components used for advanced imaging and how they fit together to meet the requirements of imaging use cases, you must understand the typical lifecycle of a document in an imaging system. This lifecycle usually consists of two phases: The *precommittal* process and the *postcommittal* process.

1.4.1 Precommittal process

From a document perspective, the precommittal process deals with the steps that take place *before* the creation of the document in an ECM repository. This process is handled by the Datacap software.

At this stage, the document does not yet exist in the enterprise repository. Therefore, its processing by business users, who use the business process management infrastructure, has not started. The precommittal process is driven by the Datacap workflow and generally includes the following tasks:

- Scanning
- Image processing
- Separation and classification of images
- Data extraction
- Data validation
- Preparation for indexing

In the precommittal phase, the documents are *collections (or batches) of independent pages*. They cannot be manipulated and used as documents in the conventional sense by business users yet.

The pages are raster images straight from the scanner or, if they are imported, native electronic documents. Their format is not necessarily the one that you want to store in the ECM repository. Instead, they are in a format that is typically most appropriate for image processing, for character and bar code recognition. You might want to convert the format to a different format for long-term preservation or other needs.

During the precommittal phase, and depending on the requirements of your imaging operations and volumes, you typically dedicate Datacap users to this process for efficiency reasons. (These users are technically *not* ECM users.) They handle document scanning, data entry, verification, and processing exceptions.

The precommittal process must be as short as possible, occurring in minutes or hours at the most. The short time frame is necessary because the business document, as known by the ECM system, has not been created yet. Therefore, it has not been officially received and has no date of record, apart from the possible date stamp that might be used to endorse the document at scan time. From a business perspective, it is not optimal if the precommittal process takes too long, especially if it means delaying revenue. The longer it takes for the documents to progress through the precommittal phase, the more the start of the associated business processes is delayed.

1.4.2 Postcommittal process

The second phase, known as the *postcommittal process*, starts with the creation of the document in an ECM repository, such as IBM FileNet Content Manager. It spans the entire lifecycle of the document up to its disposal. During this phase, all of the functions of FileNet Content Manager, including content and business process management, can be used to the fullest extent.

At this stage, the document is a conventional ECM document. It consists of one or more pages, with searchable metadata that has been extracted during the precommittal processing phase. The document is classified based on the document types that are defined in the content repository. It is typically filed in a folder structure that meets the business requirements. People can search for documents or accessed randomly while browsing. More typically in an imaging application, the documents are distributed to business users through work items that are circulated through a workflow as part of a case, for example if the organization uses IBM Case Manager.

Documents can be associated with a FileNet Content Manager workflow in one of two ways:

- The document initiates a new workflow instance upon entering the FileNet Content Manager system. The document gets attached to the new workflow.
- The document reconciles automatically with an already running workflow instance when specific conditions are met.

In addition to the document being attached to a workflow instance, the properties of the document can be transferred to the FileNet workflow or case data fields so that they can be used to automate the business processing logic. The document properties include the data that is captured during the precommittal phase by Datacap. For example, such properties can be used to evaluate routing conditions or the completeness of the data gathered or to interact with a rating or business rules engine.

In some cases, depending on business requirements, the document can be converted automatically to another format by using the FileNet Rendition Engine. It can be viewed, annotated, or redacted while it is being circulated and processed by users. Functions can be extended by adding licensing for Case Manager when additional flexibility and collaboration are needed for processing collections of data and documents that belong together and require coordination.

The ECM Administrator defines users who are involved in the postcommittal business process as ECM users. The process extends across the entire lifecycle of the documents, possibly over many years, until they are purged from the system based on business requirements, rules, and regulations.

Figure 1-1 shows an example of advanced imaging process.

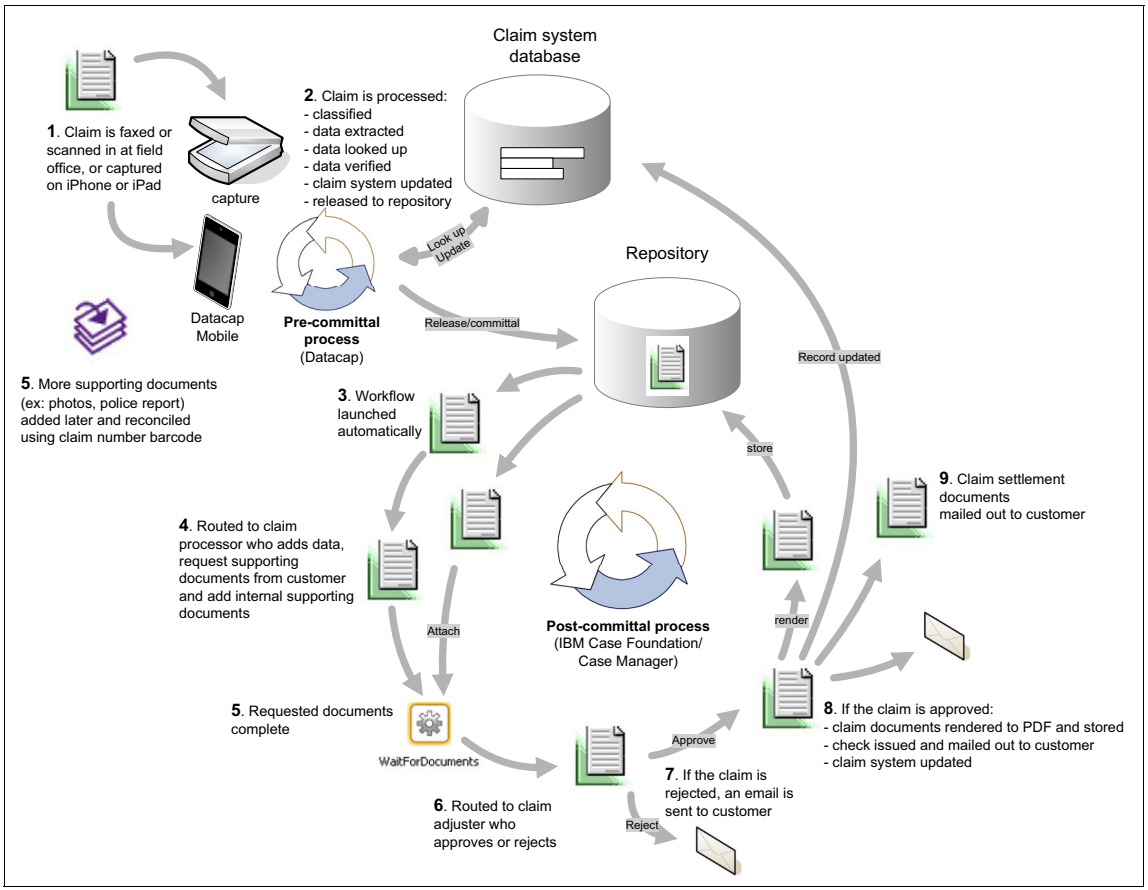


Figure 1-1 Example of an overall production imaging process

1.4.3 New possibilities blur the boundaries

The architecture of Datacap supports use cases where documents are processed less linearly than described previously.

For example, as shown in Figure 1-2, you can ingest (add) a batch to FileNet Content Manager with minimal data recognition and processing, so the documents are available to ECM users as quickly as possible. At the same time, task execution continues in the Datacap workflow, such as a user-attended operation or waiting for conditions to complete. Upon completion of the outstanding tasks, Datacap automatically updates the documents that were previously ingested into FileNet Content Manager with additional metadata.

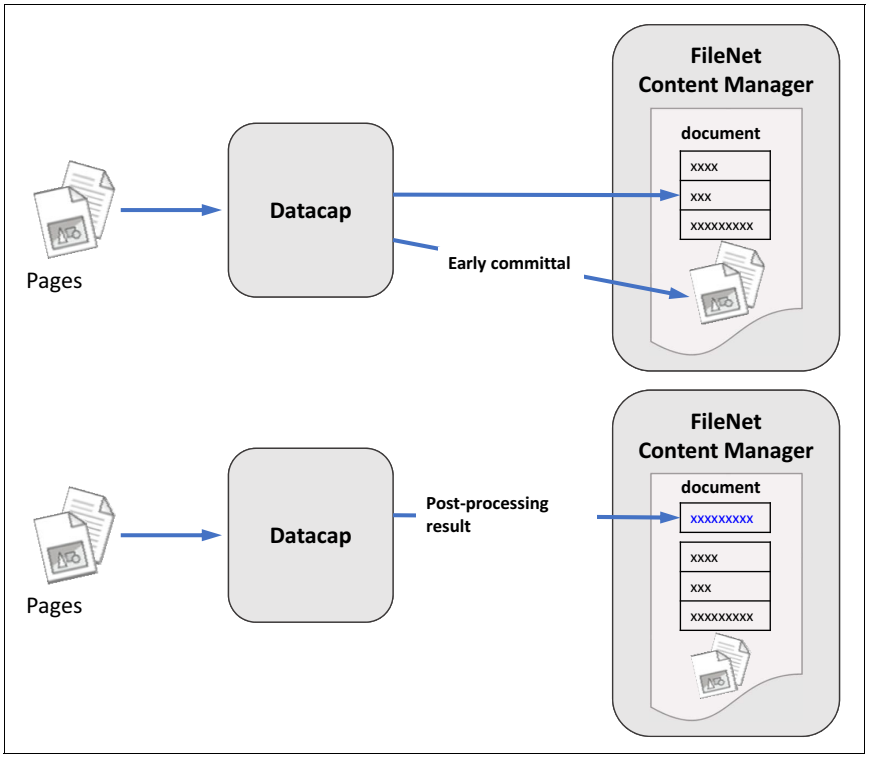


Figure 1-2 Nonlinear processing showing postcommittal indexing by Datacap

In another scenario, shown in Figure 1-3, documents are ingested into FileNet Content Manager first, starting a workflow. At specific steps in that workflow, Rulerunner actions are started to extract additional data from those documents. Similarly, existing documents might become part of a new set of tasks, for example when refinancing a loan, which requires new or additional information to be extracted from old documents. So, data capture might occur at various points in a document's lifecycle.

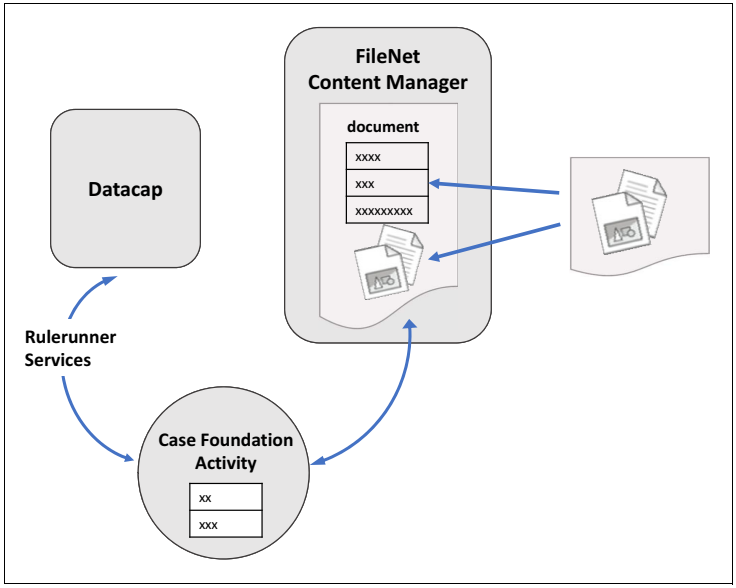


Figure 1-3 Nonlinear processing showing the use of Rulerunner

In a final example of non-linear capture and digitization, shown in Figure 1-4, documents are ingested into FileNet Content Manager, starting a workflow. At different times throughout the workflow, the Datacap application receives additional, supporting documents, such as proof of income or an identity document, which are scanned (if paper-based), indexed and attached to the existing workflow, possibly triggering new tasks and alerting ECM users to their presence, for example in an advanced case management solution.

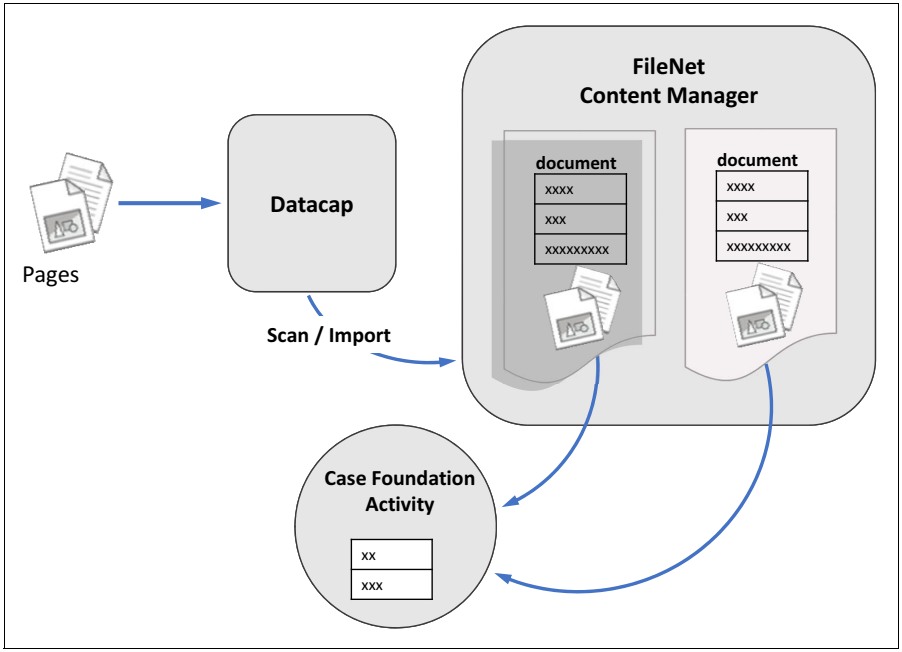


Figure 1-4 Nonlinear processing showing the use of Datacap scan or import

1.5 Examples of applications

This section reviews the types of applications that take advantage of Datacap and advanced imaging technologies.

Datacap can be used in many scenarios, from simple to complex, from handling simple correspondence documents with no structure, to complex forms with intricate layouts, such as those found in the healthcare industry. The more complex the data structure and the greater the information density, the more specialized and complex the application needs to be. This also translates into more resource-intensive processing.

Central to the implementation of any Datacap solution is the concept of an *application*. An application unites a set of Datacap capabilities with the aim of solving a specific business need. By combining a set of functions that are ready for immediate use, Datacap can address various use cases. Its sample and add-on applications, which are geared at addressing specific business situations, are not necessarily confined to those domains.

For example, consider the function that is implemented in Datacap Accounts Payable to process invoices with line items. You can use this function in other domains where there is a need to process documents with similar characteristics, such as in transportation and logistics, with shipping manifests, or in manufacturing with nomenclatures.

However, keep in mind that the reach of Datacap applications extends beyond the pure document capture aspect. Combining them with the flexibility offered by Datacap's web deployment options and the repository and business process management infrastructure of FileNet Content Manager provides a true enterprise-wide imaging solution that can use resources anywhere in the organization.

1.5.1 Cross-industry: Automated forms processing

You can achieve significant savings by automating data entry with forms-processing software. Datacap reduces or eliminates expensive typing of data and delivers data seamlessly to your business applications quickly, accurately, and more cost-effectively than manual methods.

Datacap applications apply various technologies to locate, extract, and validate data from several forms, including health claims and tax returns. This function also applies to unstructured and semistructured forms, such as invoices, shipping bills, and explanations of benefits.

Datacap can capture handprint, machine print, check boxes, and bar codes, including combinations on the same document. The dynamic and reusable rules of Datacap provide a high level of flexibility over every aspect of the capture process. They can be used across multiple types of forms to normalize data and ensure consistency. You can choose from preconfigured application building blocks and assemble them with FastDoc, the point-and-click Datacap rapid application configuration tool. Or by using Datacap Studio, Datacap's advanced configuration environment, you can follow a more low-level approach by selecting from hundreds of prebuilt actions in the action libraries, modifying them, or creating new ones. You can build complex forms processing workflows without expensive programming and test and implement them in a short period of time.

1.5.2 Cross-industry: Distributed capture

Significant savings in shipping costs can be achieved if you are able to scan documents or capture them using a mobile device at the point of origin and send them electronically. This same concept applies if your operators in charge of verification can work from anywhere with a workstation and a browser, whether at home or in a lower-cost area. Reduced cost, speedier input, and more IT flexibility for your organization are among the benefits of distributed capture.

Datacap Navigator is thin-client capture software that enables browser-based scanning and verification or indexing, which greatly reduces implementation and administration overhead. These thin clients provide integrate with your back-end business applications and image repositories, and they are easily assimilated into your existing environment, so they reduce expenses without disruption.

One key to the success of a distributed capture solution is comprehensive oversight by an administrator. To this end, Datacap Navigator provides administrative tools to monitor all work that is done remotely. A user logs in to Datacap with a password and starts working. All user permissions and privileges are centrally controlled by the administrator.

1.5.3 Cross-industry: General business documents processing

Virtually any business can benefit from the automated classification and data capture technology provided by Datacap to reduce costs and improve document processing time for various back-office documents. These documents include the following examples:

- Inbound sales orders and subscriptions
- General business correspondence
- Human resources documents, such as job applications, résumés, beneficiary statements, withholding forms, reports, and contracts
- Marketing documents, including free-form documents, data sheets, product descriptions, press releases, announcements, and white papers

One way to achieve results quickly when processing various document types is to use the flexible capture capabilities of Datacap. With minimal configuration, you can configure the document types for your structured and semistructured information. Datacap then automatically locates data and assists you with adding new document types with a few clicks as you go.

Datacap relies on a unique feature called *fingerprinting* to identify incoming documents based on their layout and match them to known document types.

After a document is identified, the data is located and extracted automatically. In most cases, no operator needs to get involved.

However, in some cases, the document is of an unexpected type, or there is so much variability that its type cannot be determined with high confidence. In such cases, the document must be processed manually by an operator who determines the type of document that is being processed. The operator visually locates and points in the image to the key data that defines the document type. Examples of such data include an invoice number, purchase order, or vendor name, as in the case of an invoice.

From this manual processing, Datacap can record the recognized data and its location on the image. It then uses this information to automatically recognize and process similar documents in the future. Over time, the exceptions become less frequent. You do not need to go through an extensive, time-consuming training and setup phase to process your documents. Instead, you can start document capture quickly and improve the process as you go, which is called *flexible capture*.

The Flex sample application, together with the Flex Manager configuration tool, provide everything you need to set up your flexible capture operations quickly. This layout is populated automatically with the appropriate fields and corresponding image snippets that are based the type of document that has been recognized. For text-intensive, free-form documents, Datacap uses the IBM Content Classification Module to classify and separate documents automatically. It eliminates the need for tedious manual prescan preparation, which enables you to process batches that contain multiple types of documents.

Datacap includes several application *templates* that help you build applications quickly and efficiently, depending on the types of documents you want to capture. See Chapter 6, “Structured forms application” on page 131 and Chapter 7, “Unstructured document application” on page 147 for more information.

1.5.4 Cross-industry: Accounts payable

Organizations that process invoices manually or scan without the help of optical recognition technology can significantly increase efficiency and accuracy through recognition-assisted automation.

Datacap Accounts Payable is designed specifically for processing invoices. The application automatically extracts all of the important information from an invoice, including line items, and delivers it to your business application with no manual data entry.

Datacap Accounts Payable includes special invoice-centric functions. It is designed for the unique requirements of Accounts Payable processing. It includes the ability to apply global rules to all or specific types of invoices, run multiple rules per field, automatically find fields and line items over multiple pages, reconcile purchase orders, and attach vendor numbers. Figure 1-5 shows the user interface of Datacap Accounts Payable.

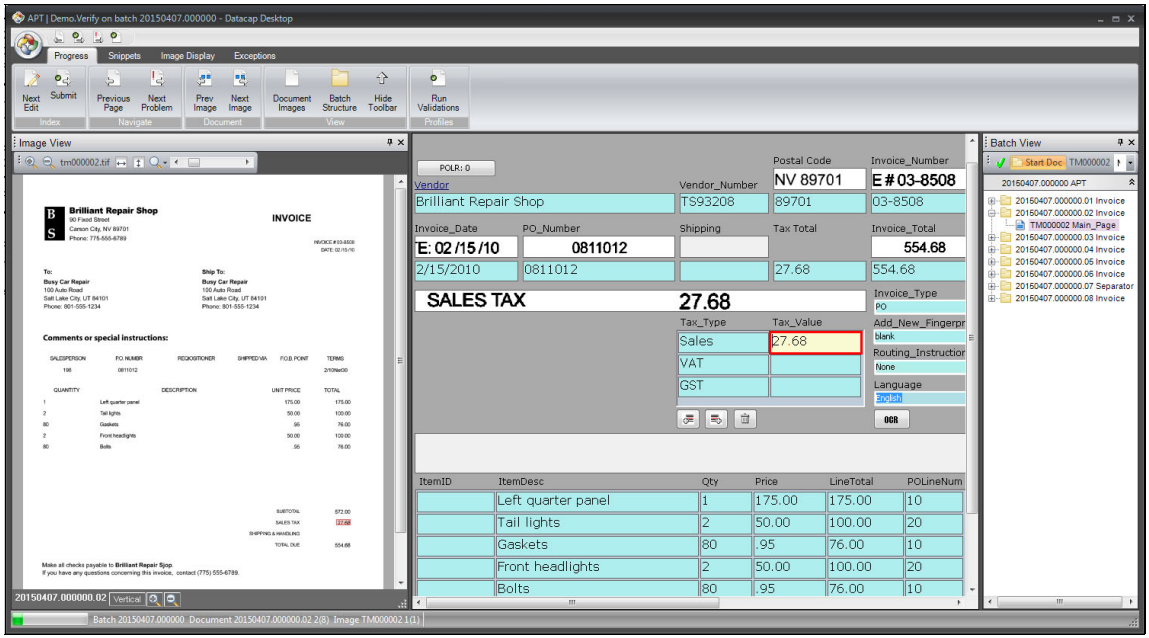


Figure 1-5 Datacap Accounts Payable add-on application for capturing invoices

The Datacap Accounts Payable add-on application includes the following features:

- Processes new invoices dynamically
- Automatically attaches the vendor ID number to known invoices
- Supports multipage invoices and line item capture
- Supports multiple languages simultaneously
- Looks up data and checks math to ensure accuracy
- Provides plenty of flexibility to configure the rules to manage your invoice
- Formats data to feed into Accounts Payable systems
- Provides for easy purchase order reconciliation at data entry
- Enables you to notify the system administrator by email that a new vendor needs to be added to the database

Licensing: Datacap Accounts Payable is an add-on application that requires additional licensing.

1.5.5 Cross-industry: Surveys

Although many processes adapt easily to the electronic environment, survey and opinion processing has stubbornly remained a paper-based process. Part of the reason is that a survey is not valid until it has been completed. Also, many in-depth surveys are too long to fit easily into an online form. If a survey taker abandons the survey at any step along the way, the survey data becomes useless.

Datacap helps marketing companies acquire and process data faster and at lower cost by reducing processing time. Datacap's ability to handle various unstructured forms enables maximum flexibility in survey-form design.

By using optical mark recognition (OMR) technology, Datacap captures completed check boxes or bubbles, machine print data, bar code data, and handprint commentary or explanations. It interprets the values and uploads them, together with information about the respondent, into a survey database for analysis.

For convenience, surveys can also be scanned or verified in a browser with Datacap Navigator. The thin client delivers the same functions for onsite or remote data gathering for conferences, trade shows, and mobile surveyors.

An example of a survey application is shown in Figure 1-6. The documents can be as diverse as questionnaires, surveys, tests, evaluations, time sheets, applications, lottery forms, and inventory counts.

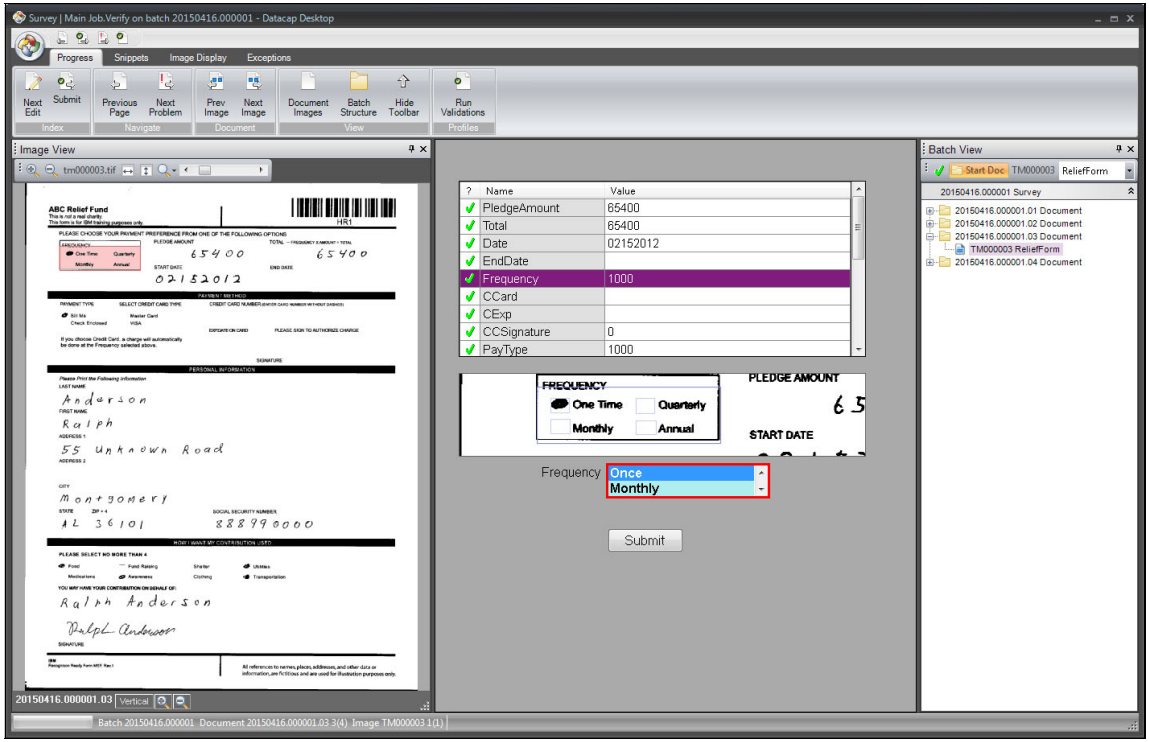


Figure 1-6 Sample application to capture of surveys

1.5.6 Government: Tax return processing

Tax and government revenue organizations rely on document imaging technology to accelerate tax data input, enable easier access to information, streamline operations, and provide better service to taxpayers. Essentially, they can shorten the turnaround time from the receipt of a return to the issuance of a refund check. Although federal, state, and local governments encourage businesses and individuals to file returns and other documents electronically, a significant volume of tax forms still arrives at mail centers in paper form.

Datacap coordinates the scanning of tax documents and attachments; the extraction, validation, and hand-off of data to back-end systems; and the indexing of images into repositories with minimal human intervention.

Datacap Navigator can enable remote scanning or remote verification by at-home workers to help government tax departments increase productivity and distribute work to lower income areas.

What makes Datacap the solution of choice is its ability to handle exceptions and problems that are always associated with high volumes of forms, especially hand-filled forms such as tax documents.

The Datacap 1040EZ sample application demonstrates the processing of US tax return forms. The application uses ICR technology to capture hand-written characters (Figure 1-7). It validates the data by checking the presence of mandatory values, applying calculations, and enforcing verification against tax rules.

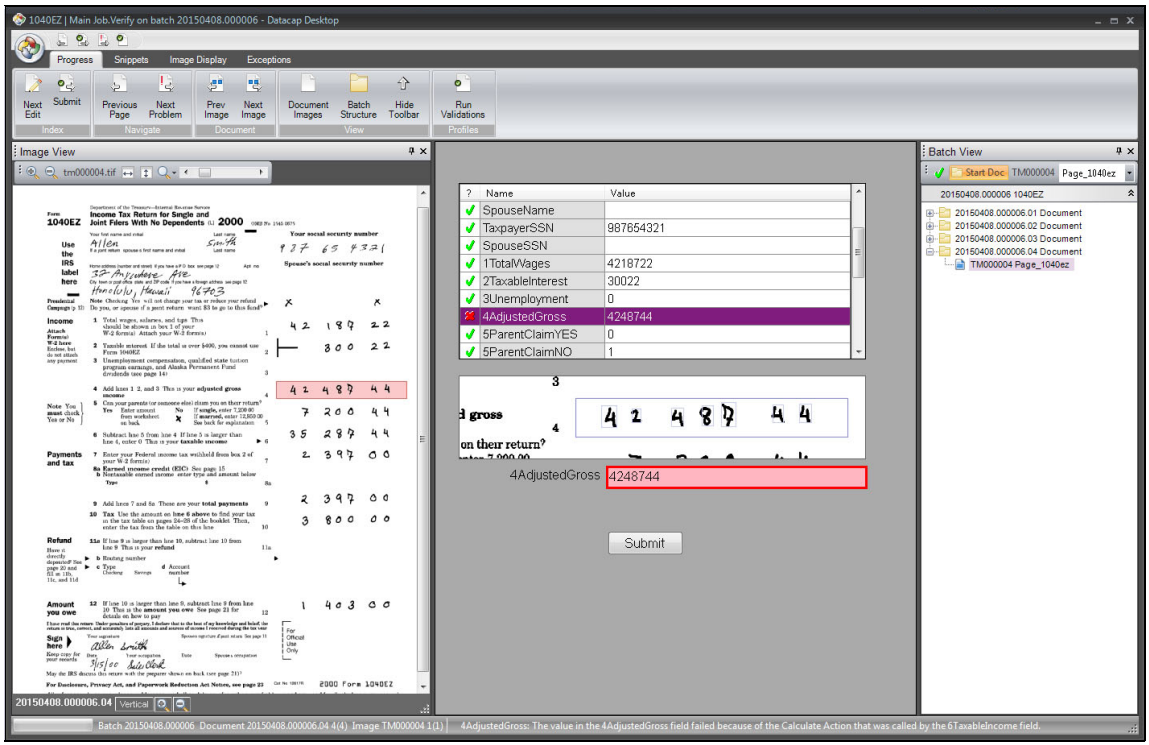


Figure 1-7 Sample application for capturing a US tax return form

1.5.7 Healthcare and insurance: Medical claims

Insurance companies were among the first to embrace scanning and document management to help control costs and streamline their operations, which depend on the ability to rapidly and accurately process claims and policies. They also need to ensure compliance with Health Insurance Portability and Accountability Act (HIPAA) regulations regarding privacy and data standards.

Datacap Medical Claims is a solution that automates data entry for CMS-1500 and UB-04 medical claim forms used in the United States. The application helps to eliminate costly, error-prone manual data entry and accelerate claim processing time.

Datacap Medical Claims manages the entire capture process, from the scanning of claims to the recognition of data fields to the validation and verification of data for accuracy. It also coordinates the upload of HIPAA-compliant claim data to adjudication systems for payment.

Because Datacap Medical Claims (Figure 1-8) is built on the Datacap platform, health insurers can take advantage of browser-based distributed scanning and remote indexing for more efficient distribution of work.

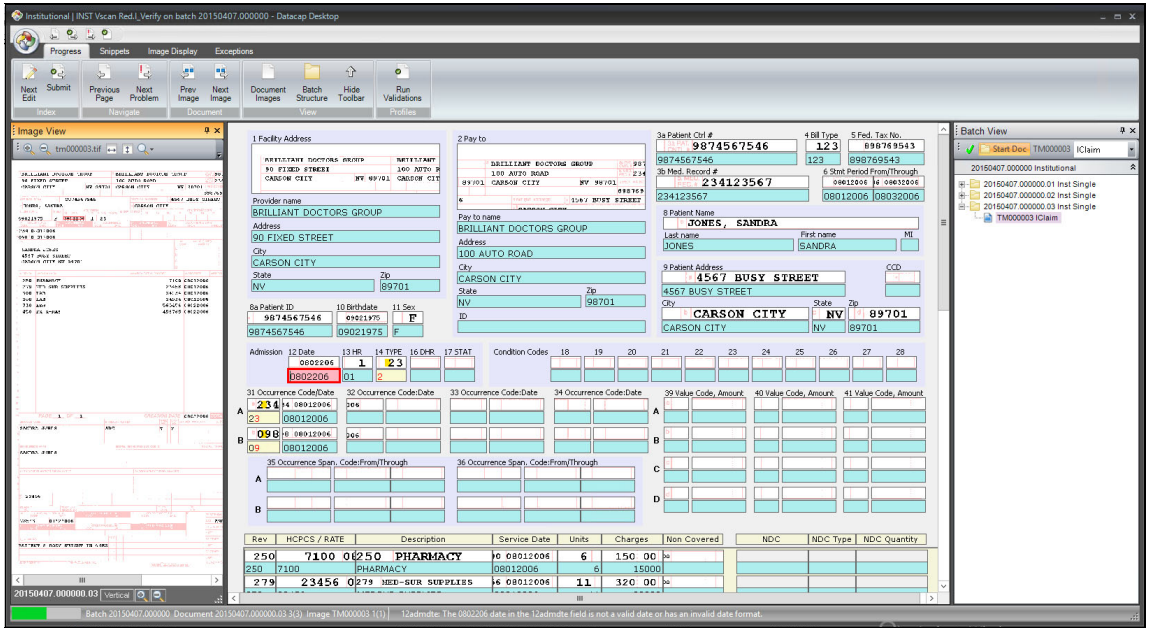


Figure 1-8 Datacap Medical Claims add-on application showing the capture of healthcare forms

Datacap Medical Claims has unique features that claim processor value. For example, it performs automatic database lookups to validate data against provider, member, diagnoses, and procedure codes. It also has an intuitive verification interface that enables fast and easy identification of claim data.

Because of the high data density of medical forms, Datacap uses the color dropout technique to remove grid lines in the form background to make it easier to recognize useful data. Datacap achieves this processing by using special forms with red ink that drop out when scanned with color filtering or image processing in the scanner.

Figure 1-9 shows the original form on the left side and the color-dropped-out form on the right. The form on the right side shows only the data that is pertinent to the capture application.

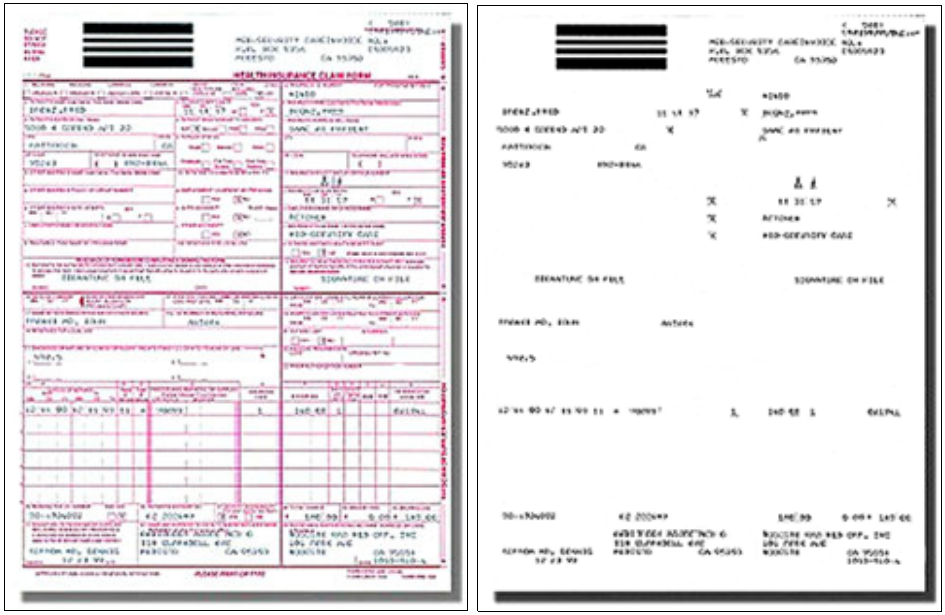


Figure 1-9 Color dropout to remove grid lines

Datacap Medical Claims includes the following features:

- Generates a dynamic template for every page
- Supports all CMS (Centers for Medicare and Medicaid Services) forms and variations with no preconfiguration of templates
- Offers the ability for users to capture 100% of fields from both CMS 1500 and UB-04 claims.
- Captures and stores attachments with an accompanying claim

- Supports permanent image overlay for archival
- Offers advanced validations to ensure accuracy, including lookups for member, provider, diagnosis, procedural terminology code, place of service, service date check, and math calculation on charges
- Enables configuration and modifications without expensive programming
- Offers HIPAA-compliant 837 EDI export

Licensing: Datacap Medical Claims is an add-on application that requires additional licensing.

1.5.8 Banking and finance: Loan applications

Mortgage loan applications can total hundreds of pages of different document types, from titles and credit reports to appraisals to certificates of occupancy. The faster a financial institution can process all the documents required for a mortgage, the faster they can provide funds to the customer. Because mortgage documents must be maintained for the life of the mortgage and beyond, fast and efficient imaging can deliver cost savings and help maintain compliance.

Datacap automatically captures data on loan documents to feed back-end systems and accelerate indexing for image storage and retrieval. With the help of content-based classification technology, Datacap helps classify and identify all of the different documents within a loan portfolio, which reduces or eliminates the need for tedious manual prescan preparation.

By using the browser-based Datacap Navigator application, financial institutions can distribute scanning to branch offices and affiliates so that loan applications can be scanned seconds after being signed by the customer.

1.5.9 Transportation and logistics: Shipping documents

Transportation and logistics post unique challenges in the gathering and management of data. Transportation companies have a mobile workforce and often a distributed sales force. Tracking deliveries is a paper-intensive process. Also, regulations governing the transport of goods across state lines and between countries are placing increasing constraints on the business.

Companies can use Datacap Navigator or Datacap Mobile on iOS or Android devices for distributed remote scanning of documents generated in the field, such as proofs of delivery, sales orders, and fleet maintenance documents. By using Datacap this way, companies can reduce the enormous cost of mailing or faxing documents to a central headquarters for processing, accelerate input, and eliminate delays in billing.

Shipping and logistics companies face growing pressure to provide complete information about the contents of a shipment when the goods cross the border between states or countries. The US Patriot Act and other regulations demand rapid and accurate data extraction from complex commercial invoices to fulfill customs requirements. Datacap Accounts Payable, with its ability to capture line items from multiple page invoices, is well-suited for this use (see 1.5.4, “Cross-industry: Accounts payable” on page 19).

2

Advanced Datacap capabilities

This chapter focuses on the advanced imaging capabilities and functions provided by IBM Datacap. Concepts and capabilities are introduced as part of a typical Datacap process, from document acquisition, to data capture, to committing contents and data to enterprise content management (ECM) repositories.

This chapter includes the following sections:

- Functional overview
- Multichannel input
- Transforming documents into actionable data
- Delivering documents and exporting data
- Datacap user interfaces
- Application configuration

2.1 Functional overview

The purpose of Datacap is to acquire documents, extract useful information from them, and feed them into other business processes downstream. Its strength is its ability to perform these tasks with a high degree of automation, flexibility, and accuracy.

At a high level, Datacap functions can be organized into three areas:

- Acquisition of documents from several sources
- Processing of documents to extract useful information
- Delivery of content and data to back-end systems

These functions are integrated into a task flow that controls the processing of the documents from acquisition to delivery, with background tasks whenever the processing can be automated, and with foreground tasks when human interaction is required, such resolving errors and ambiguities in the extracted data.

Datacap handles the following main functions:

- Acquires paper documents from scanners, multifunction printers, or mobile devices, such as smartphones and tablets
- Imports electronic documents or existing images from a file system, fax, or email server
- Cleans up images and prepares documents to improve data extraction with image-processing capabilities, such as deskewing, removing lines, smears, and borders
- Classifies and separates document based on type to determine which data needs to be extracted
- Extracts data by using recognition technologies:
  - Optical character recognition (OCR) for machine-printed characters
  - Intelligent character recognition (ICR) for handwriting, typically detached block letters, but also cursive writing on checks or in other well identified contexts
  - Optical mark recognition (OMR) for identifying checked boxes and other marks, such as bubbles in surveys or a signature on a form
  - Bar code reading of several types, including one-dimensional bar codes, such as those used for price reference in stores, or two-dimensional bar codes that are used to encode much larger sets of data, such as name, address, or shipping information
- Checks the accuracy of extracted information and corrects errors against business rules

  Datacap can also automatically look up information in a database from the partially recognized data. It can trigger verification and validation by a human operator when confidence in the data accuracy is below a set level.
- Learns automatically from the experience of human operators and the processing of documents to improve accuracy over time
- Exports image documents and extracted data to FileNet Content Manager or other ECM repositories, databases, or business applications
- Organizes the flow of tasks in the capture process from scan to export, including handling of exceptions, into a workflow
- Controls access to the system and tasks by using functional security
- Monitors progress of capture operations and fixes problems in real time
- Reports on capture operations and provides statistics about how well the system is performing

- Supports flexible deployment scenarios, from central mailroom-type operations to distributed imaging over the web and mobile devices, to regional and branch offices using multifunction printers and the distributed deployment capabilities of FileNet Content Manager

2.2 Multichannel input

Datacap can acquire documents from the following sources:

- High-speed production scanners
- Multifunction printers
- Remote desktop scanners
- Mobile devices
- Fax servers
- Email servers
- File systems

2.2.1 High-speed production scanners

High-speed production scanners are found in large mailroom operations. They can scan hundreds of pages per minute and sustain high ingestion rates throughout the day, running in the thousands of pages. This way, they achieve a low per-page cost for large digitization needs.

Typically, the scanners are connected to powerful workstations by using an ISIS driver and are served by dedicated operators and, depending on volumes, a team of people to prepare the paper documents manually for scanning. They remove documents from envelopes, remove wrinkles and staples, sort the pages that go together, and so on.

Next, documents are assembled into *batches,* based on how they are processed later. For example, loan application forms and their supporting documents might be prepared so that they flow in a predefined order that is repeated within a batch.

In another example, when a batch has high variability and little structure, it is useful to insert document separator sheets with a bar code to mark document boundaries. Separator sheets can contain check boxes or other printed data to facilitate the classification process. You can use separator sheets to split a large batch into smaller ones or into separate documents, and you can use a batch cover sheet to automate the indexing of data that is common to all documents in a batch.

In most cases, the total number of pages and documents that have been prepared per scan run is noted. The scanner operator checks that the number of scanned pages and documents matches the physical batches of paper and correct errors if necessary.

It is important to rememmber that in this type of centralized scanning operation, operators are not subject matter experts on the documents that they are manipulating and are often given incentives based on volume of documents processed. Speed, high volume, and efficiency per operator are the key metrics.

2.2.2 Multifunction printers

Multifunction printers (MFPs) that also copy and scan are frequently shared by many users in offices. The scanning by an MFP is performed by a single individual in the context of completing a business transaction, such as an account opening, a loan application, or an insurance claim, at a branch office. Each scan operation typically involves only a few documents. However, in the aggregate, this can be a high volume from a Datacap system perspective.

Multifunction printers are connected to a dedicated MFP server that provides secured connectivity and serves the Datacap user interface that the user sees on the device. The MFP server acts as a front-end to the Datacap server. It routes all of the documents being captured at the various MFPs in an office.

Unlike mailroom operations, office users are typically subject matter experts in their business area, but not necessarily well-versed in scanning and imaging. They just need to capture paper documents in the course of their business duties, such as processing a claim. MFPs provide a simple user interface where users log in and select among preset types of business transactions, specific to their roles, and input the requested documents, which are then scanned directly and securely to Datacap without compromising personal and other sensitive information. Because the users are subject matter experts, they are often able to enter or select simple indexing information based on customer or policy numbers (assisted through lookups) and spot and correct errors immediately.

MFPs also remove the need to print and send critical documents between offices by courier, which eliminates the associated cost of handling physical documents. This accelerates business processing by removing the extra step of sending documents to a central location to be queued for scanning. Therefore, processing takes minutes rather than days.

2.2.3 Remote desktop scanners

Personal desktop scanners are typically used in highly distributed scanning operations at remote offices or by users who work from home. These low-volume scanners are driven through a TWAIN interface, often using the Datacap Navigator user interface. For example, while opening an account, this enables employees at a front office to capture, in real time, the photo IDs and pay stubs while customers are at the remote site. Employees do not need to leave their desks and can also key in additional information while chatting with the customers. This is a convenient and easy way to capture critical information at the earliest entry point in the business process without needing any infrastructure other than network connectivity.

2.2.4 Mobile devices

Mobile devices such as the iPhone, iPad, and Android phones and tablets can be used to capture documents in the field where there is no network infrastructure other than the one provided by the cell phone service. For example, a claim adjuster can use an iPhone or iPad to capture signed claim information along with photos of a damaged vehicle at a claimant's home. The adjuster can also capture the bar-coded Vehicle Identification Number (VIN) on the vehicle to compare it to the one on record.

In other instances, such as in warehouses, workers who do not have desks and are constantly on the move checking inventories need the capability to capture product labels, bar codes, and so on.

Datacap mobile capture enables users to digitize documents directly into Datacap applications. Its intuitive user interface enables them to snap or import photos of one or multiple pages of a document and automatically correct any optical distortions that result from taking photos at an angle. They can also reorganize the pages and enter indexing information or capture bar code data to index fields. Multiple documents can be captured and accumulated locally in case of poor cell coverage or when users prefer reviewing the documents before they are sent to the Datacap server.

2.2.5 Fax servers

Fax communication has gradually been replaced by email, but there are industries where it is still critical because of the large fax infrastructure in place or extensive business processes that have been built on it over the years. In other instances, fax documents are still the only electronic documents that are accepted as legal proof rather than the physical document.

Datacap can ingest and process faxes from OpenText RightFax. It integrates directly with the fax server's APIs to poll periodically for incoming faxes. After they are retrieved, faxes are processed by the Datacap server in the same way as any other image. Each fax makes up a document in the Datacap batch. In addition to the pages, fax header information is retrieved from the server and placed into document variables, such as the fax ID, number of pages, originating fax number, name, and telephone number.

2.2.6 Email servers

Email has essentially supplanted fax for exchanging business documents. This process has accelerated dramatically in the last few years, especially for transactions between customers and businesses, because email is generally free of charge and has become so ubiquitous. In business-to-business information exchanges, even in industries such as insurance that have been great users of fax technology, there has been a shift to using email messages, sometimes with hundreds of attachments, to transfer files.

Datacap can ingest and process email messages and attachments from Microsoft Exchange and any IMAP-enabled server, which means virtually any email system, including IBM Notes software. It connects periodically and checks inboxes for new email messages, up to a configurable maximum, and filters for specific types of attachments. Each email message and its attachments make up a document in a Datacap batch. The structured email fields, such as To, CC, and Subject, are captured for indexing. Each email body and attachment page is paginated and converted to TIFF for processing in the same way as any other image document.

2.2.7 File systems

Capturing documents from a file system is the easiest way to ingest documents in a capture system. It enables you to process files of existing documents, to decouple scanning operations from downstream processing if you are subcontracting digitization to a service bureau, to automate periodic polling of machine-generated documents, such as statements, to interface with an unsupported fax server, and so on.

Datacap can monitor a directory structure in a Microsot Windows file system, or any remote file system that is mounted on Windows, for batches of documents to import. Any file type is supported, and multiple processes can monitor the same directory structure to offer scalability and high availability.

Datacap can also retrieve and prepopulate imported documents with index information that is provided in a companion XML file. Then, for example, a service bureau can supply basic indexing information, along with the scanned images, to be verified and complemented in-house with data extracted downstream in Datacap.

2.3 Transforming documents into actionable data

After acquiring documents, Datacap transforms their contents into actionable data for use in business processes. This function is at the core of Datacap and puts into action a wide range of technologies and techniques to represent and manipulate document components, make them more amenable to recognition, isolate and extract relevant data, and assure quality.

2.3.1 Document organization and taxonomy

To process documents, Datacap needs to have a way of representing them so that it can inspect their structure and component parts, manipulate them, and apply its processing to extract the target data. In fact, it needs to represent not just the documents but the entire physical content of an "acquisition session," that is, the set of documents that must "travel together" and that you want Datacap to handle as a whole.

For example, from a business perspective, you need to have all documents that pertain to a credit card application (such as the signed application form, a pay stub, and a photo ID) processed by Datacap as a single transaction so that they can be delivered to business users together. You also need to uniquely identify that transaction for tracking purposes so that business users can refer to it and retrieve it from the back-end ECM system.

For this purpose, Datacap has a flexible object model called the Document Hierarchy or Datacap Object (DCO) that comprises a *batch*, or a container and unit of work that is processed as a whole. It contains one or more *documents* with one or more *pages*, each with one or more *fields*.

The term "batch" is used because it corresponds to the physical grouping of individual sheets of paper or pages to be acquired together. The *document* is more of an abstract notion and is actually determined by Datacap *after* inspecting the content of the batch and applying rules to separate documents. For example, this could be finding separator pages or known page types that mark the start or end of documents or simply recognizing that the batch is structured in a way that every four pages marks the start of a new document.

Fields can be associated with the Document Hierarchy at any level and are used to store data extracted by Datacap or entered by users. For example, the Employee Name, Social Security Number (SSN), and Net Pay fields can be associated with the pay stub page to store the data extracted by Datacap from matching zones on the captured image. When a bank's agent captures the documents for a credit card application by using an MFP, a unique credit card application number is assigned automatically by the bank's credit card system. It is added to the Application Number field that is associated with the batch that contains the three application documents. The data can then be used to index the documents when they are exported to the ECM system. See Figure 2-1 on page 35.

The Document Hierarchy is a Datacap construct that is defined at design time, the "Setup DCO," for each Datacap application and is used as a blueprint for creating runtime instances during the capture process. It then collects the extracted data until the documents are exported to the back-end repository at which point the instance is deleted by the Datacap maintenance process.

To facilitate the configuration of the Setup DCO and ensure proper mapping of the fields used in the business process, Datacap provides the ability to import the property definitions of the document classes from a back-end repository.

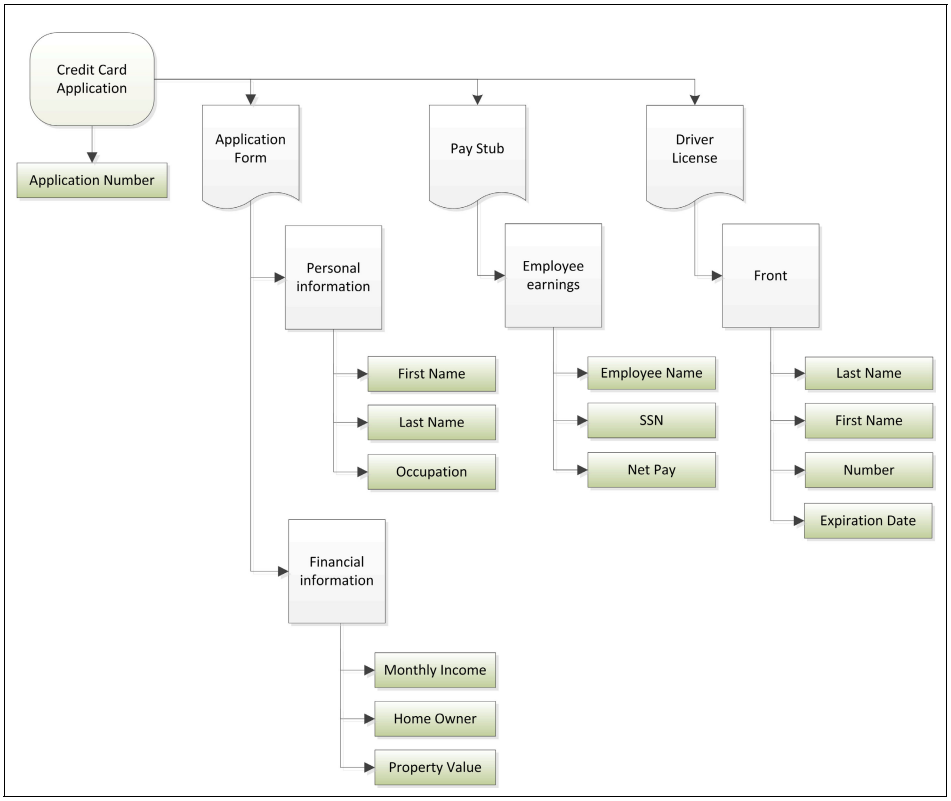


Figure 2-1 Example of a Document Hierarchy for the processing of credit card applications

2.3.2 Document processing flow

In the previous section, you learned how documents are represented by a Document Hierarchy object in the Datacap system. This section explains how Datacap processes those documents at run time, based on the structure defined in the Document Hierarchy.

As Figure 2-2 on page 36 shows, at a high level, the Datacap process begins with the ingestion of pages acquired in a batch and finishes with the delivery (release) of documents and their metadata to a back-end repository and line of business (LOB) database.

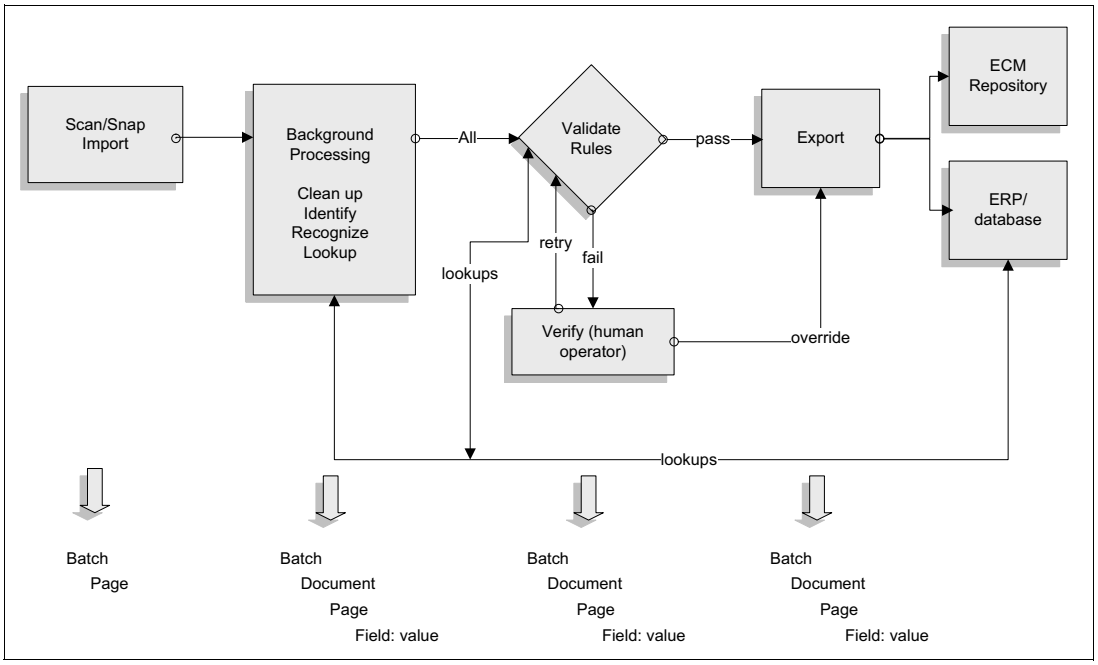


Figure 2-2 Datacap process

This process is broken down into several individual *tasks* that run automatically in the background or manually when user assistance is needed. Apart from scanning or acquiring documents with a mobile device and verifying extracted data that failed validation rules, everything else in Datacap typically runs automatically.

Ingestion from input channels

The first task in a Datacap process creates a *batch,* or more concretely, a working directory on the Datacap file server where all the images produced by a particular input channel are stored and processed together. This also creates a runtime instance of the Document Hierarchy based on the Setup DCO defined for the application, which then gets updated at each step of the processing flow until the documents are exported to the back-end repository.

Core document processing and data capture tasks

Datacap processes the batch automatically based on a set of instructions or *rules* defined for each task. The processing rules are run in a predefined order on the various components of the runtime Document Hierarchy: batch, documents, pages, and fields.

Many different operations can be performed, depending on the specifics of the application. Typically, they include the following operations:

- Cleaning up and enhancing the images to help the recognition process
- Identifying the class of pages by using *fingerprints,* bar codes, pattern matching, keyword searches, or a combination of these techniques, and assembling pages into documents
- Recognizing the data from the zones of interest on images and writing them to the fields of the Document Hierarchy
- Validating accuracy, formatting, consistency, and completeness of the recognized data and looking up missing pieces of information in external systems

The Verify task

The data extraction process flags data with a low recognition *confidence level* or validation errors so it can be verified by an operator. Datacap provides a Verify task that can be run in the web or thick clients for this purpose.

The Verify task presents problem fields together with their corresponding *image snippets* for the operator to manually check and update the data and submit the changes for validation against the rules associated with the fields. This process is repeated for each problem field until the data is successfully validated or possibly overridden by the operator.

The Export task

After the images have gone through the data capture and verification tasks, Datacap automatically exports the documents and extracted data to the back-end systems. The Export task includes the processing that is needed to prepare the documents (possibly converting them to another format), establish the connection with the back-end systems, and index and upload all the documents that were contained in the batch.

2.3.3 Rule processing

One of the strengths of Datacap is its ability to perform operations in the background. This section explains how the Datacap workflow and rules are processed with the Document Hierarchy.

Rulerunner engine

To run background tasks, Datacap relies on the *Rulerunner* engine and an extensive library of *rules* and *actions* assembled into *rulesets*, which are functional blocks that run on the objects in the Document Hierarchy. Actions can be invoked manually, such as when a human operator validates field values.

In most cases, however, they are invoked automatically by the Rulerunner engine, which is set up to monitor a job queue and run tasks automatically as batches move forward through the Datacap process. Figure 2-3 illustrates the process.

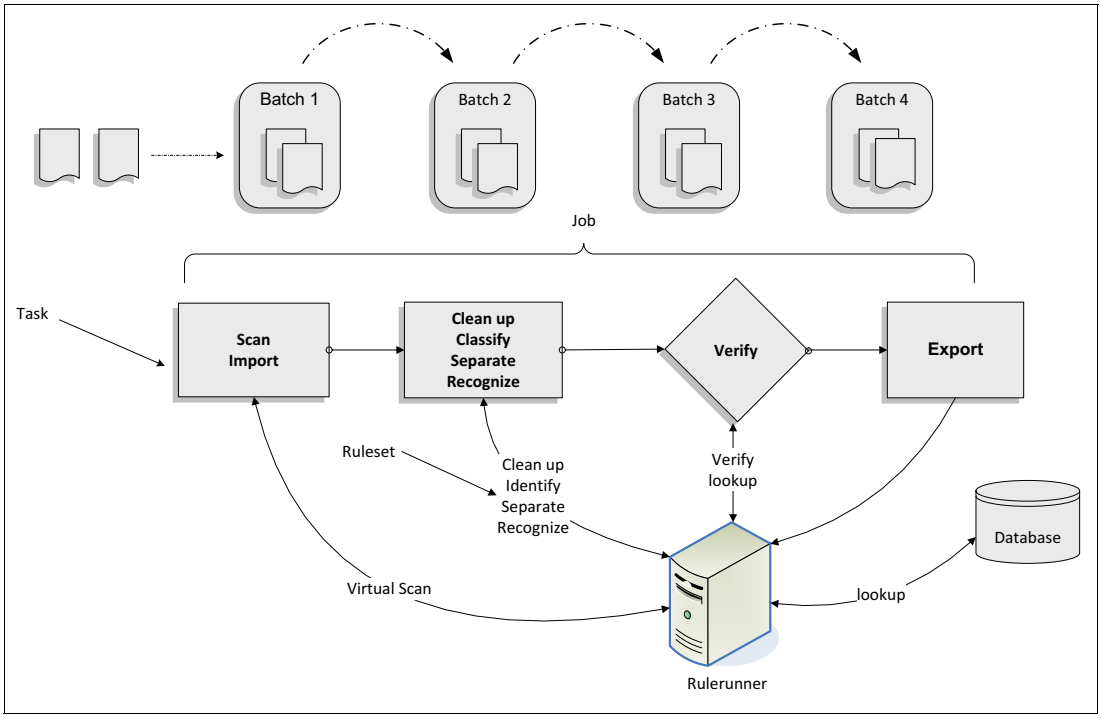


Figure 2-3 Rulerunner executing the rulesets specified in each task

In addition to the Document Hierarchy, Datacap has a *workflow hierarchy* that describes the relationship between a *job*, *task*, *task profile*, *ruleset*, *rule*, *function*, and *action*. Creating a Datacap application entails defining these two hierarchies and the interplay between them.

Job, task, and task profile

A *job* is a particular combination and sequence of discrete tasks in the workflow of a given application to address a specific operational scenario. For example, we could set up a "mailroom scan job" with specific tasks:

- Process large scan runs of credit card applications from a production scanner.
- Classify and separate documents with separator sheets.
- Recognize, extract, and verify the data.
- Export the applications to FileNet Content Manager.

We could also set up another job called "MFP scan job" with similar tasks for capturing the credit card documents from an MFP, but with tasks modified to receive the documents from the MFP server, rather than the scanner, and to classify and separate the documents without separator sheets, because each batch contains only the documents from a single application.

When a task is run, Datacap executes the *rulesets* that were defined in the corresponding *task profile*. This profile is a sort of template that is used by Datacap as an entry point to invoke a task at run time.

Ruleset

A task profile is made up of several *rulesets* that are arranged in a particular sequence to produce the desired processing results. They can be thought of as “processing building blocks” that you apply to particular objects in the Document Hierarchy.

For example, a task profile called Extract can be set up to include all the functions to capture data from the batch in one high-level task. However, the capture process within that task must be performed in a logical order. To get good recognition results, you will typically need to clean up the images first, so you will assemble a ruleset called Enhance that works at page level. It applies image-processing rules to deskew and remove smears and borders, and might adjust contrast on all the images of the batch. You then will want to set up a ruleset called Identify to run at batch level to determine the types of pages and how they should be separated into documents and to drive recognition. Next, you will want to set up a ruleset called Recognize that runs optical character recognition at page level and populates the fields associated with the pages. Also, you need a Validate ruleset to apply validation rules at field level against the data has been extracted.

Compiled rulesets

Datacap includes a collection of preassembled rulesets, called *compiled rulesets*, which are self-contained building blocks of functions that can be easily assembled into an application and configured using FastDoc or Datacap Studio. They add the following benefits:

- Reduce the expertise needed to create applications.
- Reduce application complexity by standardizing how core functions are implemented.
- Reduce the occurrence of nonstandard or poorly designed capabilities.
- Make applications more consistent and easy to understand and support.

Each compiled ruleset is a full implementation of core Datacap functions and comes complete with its own user interface to display configuration parameters and options.

Compiled rulesets support inheritance and automatic binding to objects of the Datacap document hierarchy (batch, document, page, field).

The rulesets, in their un-compiled form, can be copied and edited using Datacap Studio to be customized and extended. Compiled Rulesets are available for all major functions of Datacap, such as file import, page identification, image enhancement, data extraction, fingerprint matching, and export, and, if needed, additional ones can be developed using Datacap Studio and compiled using a Microsoft Visual Studio template project available in the Datacap Technical Mastery community of IBM developerWorks:

http://ibm.co/1MwxWxW

Rule

A ruleset groups one or more *rules*, or lower-level processing capabilities, that are bound together to the objects in the application’s Document Hierarchy. They are run on demand when Rulerunner opens or closes objects as it walks through the Document Hierarchy at run time.

For example, if they have been selected and configured as part of the application’s rulesets, the rules of the Enhance ruleset are run every time a page is opened to run deskewing and smear removal.

The rules within a ruleset run only when they are mapped to specific objects of the Document Hierarchy. In addition, they run only when the ruleset they belong to is included in the task profile being run. The execution order of rules in a ruleset is dictated firs, by the order in which the parent ruleset appears in the task profile and second by the processing sequence of the objects in the runtime Document Hierarchy.

Function and action

A rule is made up of one or more *functions*. A function consists of one or more *actions*. An action represents the code that runs a particular elemental operation on the objects of a document. A function is started in the order in which it appears in the rule. If an action fails, the function that called it exits unsuccessfully, and the next function in the sequence gets executed. If the action succeeds, the next action in the function gets executed. If all actions of a function run successfully, the rule that called the function exits successfully.

By using this approach, you can construct efficient processing rules without coding. Additional information about actions, including how to create your own *custom action*, is provided in Chapter 13, “Datacap scripting” on page 295.

For example, in a rule that is used to identify the *type* of page (“Page identification” rule), several functions can be assembled in a fallback sequence, from the most to the least processing-intensive or efficient. Each function implements a specific recognition technology.

We can set up the rule to call the functions such as these:

- Identify using fingerprint
- Identify using text match
- Identify manually

Manual identification, which is merely flagging the page for a subsequent user-attended task, is called only after fingerprint and text matching fail. If the fingerprint matching function succeeds (all of the actions in it succeed), the “Page identification rule” exits, and the subsequent functions are not run.

Processing of the Document Hierarchy at run time

When a task is invoked, Datacap recursively processes each object in the runtime Document Hierarchy. It starts at batch level and proceeds to open the first document, then the first page within it, then all of the fields on that first page, and then on to the next page, and so on, as shown in Figure 2-4. It repeats this process with the next document. As it processes each object in this manner, it calls the rulesets that are bound to it. Rulesets can be configured to run on opening or on closing the object.

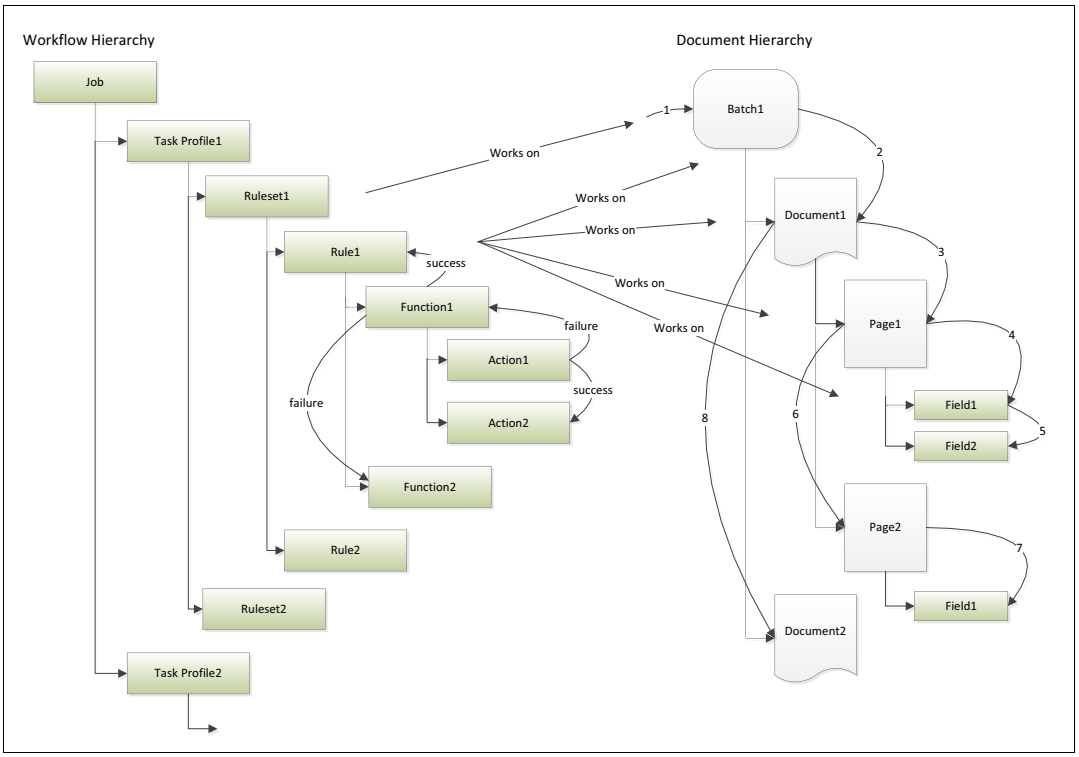


Figure 2-4 Workflow and document hierarchies and processing sequence

2.3.4 Advanced image-processing capabilities

Now that you understand how documents are represented and processed in a Datacap application, we explore the core Datacap operations run on the Document Hierarchy in more detail. We start with image processing, in this section, and then cover classification, recognition, and validation in subsequent sections.

Image cleanup and enhancements

Image cleanup and enhancement functions improve legibility and data capture processing in the later stages of the Datacap process. This filters for the following tasks:

- *Deskew* or straighten a crooked image to improve OCR performance and improve reading.
- *Rotate* an image, typically 90-degree increments. This might be necessary when, for example, certain pages that display in landscape mode were scanned as part of a batch that was processed in portrait mode.
- *Flip* an image on its horizontal axis (upside down) or mirror an image on its vertical axis (left-right), typically to correct images captured from photo slides.
- *Deshade* the image to increase crispness to better reveal text in shaded areas and graphics.

- *Dilate or erode* an image to increase or decrease the thickness of the shapes in the image without changing their proportions and to make them easier to process. This action is especially useful for character recognition when characters appear thin, with discontinuities (an “l” looking like an “i”), or conversely, as a mass that lacks details (an “i” that looks like an “l”).
- *Despeckle and reduce noise* to remove random specks or background noise that are typically introduced when the scanner sensitivity threshold is too low in black and white or by shaded backgrounds in forms.
- *Smooth* characters to repair broken segments and smooth ragged edges that occur when scanning documents printed by using dot-matrix printers.
- *Reverse* text to detect and reverse regions of the image that typically have white text on a black background, such as in table headings.
- *Remove* horizontal and vertical lines in high-density forms and tables to reduce clutter and enhance recognition of useful data.
- *Remove* the black borders that typically form on the edges of a black-and-white image that was scanned with high-sensitivity settings.
- *Remove* streaks, vertical lines, or smears that are sometimes added to an image by the scanning process.
- *Remove* blobs and punch holes when capturing pages from a ring binder.
- *Fill* line breaks to repair broken lines, as in underlined text for example, and to improve legibility for human readers.
- *Create* an outline out of shapes on the image by dropping unnecessary details to improve the legibility of a busy image.
- *Convert* an image to binary by converting color-encoded pixels to black or white pixels. This filter is optimized to reveal text in documents with dark backgrounds.
- *Apply* image “detergent” to remove noise from color images by converting the pixels in a range of similar colors to a central color value. This filter is effective for removing JPEG compression artifacts that appear around characters.
- *Remove combs* (no line on top) to drop the constraint lines used in fill-in-the-box forms without affecting the characters that have been completed. This filter is typically used before running handprint recognition.

These filters are generally used in combination, and their proper mix and settings are derived by trial and error.

In Figure 2-5, *Deskew*, *Despeckle*, and *Border Removal*, *Remove Lines*, and *Remove Combs* are selected and processed in that order. As the user selects the options in FastDoc at design time, the effect is immediately displayed at the right in most windows.

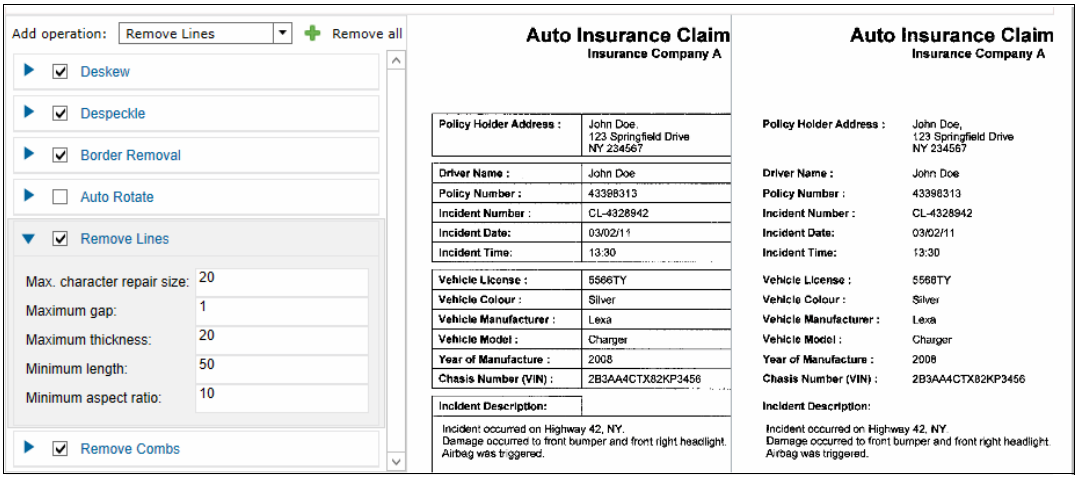


Figure 2-5 Enhancing images in real time in FastDoc

Imprinting and redaction

Datacap provides an imprint library that can be used to overlay text on an image. Alternatively, the library can be used to redact part of the image to protect personal information from public view, such as health record and social security information.

The redaction action is called in a rule attached to a target zone defined in a fingerprint to black or white it out. Alternatively, you can use text locating actions to find, navigate, and position the redaction zone over the data in unstructured documents automatically. The resulting imprinted content or redaction is flattened and burned in the image.

Typically, when redaction is used, two capture streams are implemented in Datacap. First, the redacted documents are committed to a repository with access rights for general circulation. Then the originals are committed to a secure repository with restricted access rights for authorized personnel.

By implementing redaction as part of the capture process, you can use dedicated trained personnel that can be organized to accommodate high volumes and an infrastructure of data dictionaries or lookup tables and verification rules common to the main imaging operations which helps normalize data against common business rules and reduce errors.

2.3.5 Automatic document classification

A great return on investment is derived from the ability to automatically classify documents and drive the data extraction and indexing process when ingesting documents into a back-end repository. This is one of Datacap's most critical functions.

Given the variability in the types and quality of documents that are routinely processed, it is often necessary to use a combination of techniques to reliably identify them. Datacap includes the following ways to classify documents:

- Fingerprinting
- Bar code-based identification
- Structure-based identification
- Text and pattern matching

- Identification with IBM Content Classification
- Manual page identification

Each method is described in the subsections that follow.

Fingerprinting

The most innovative Datacap feature in the area of document classification is the use of *fingerprints*. A *fingerprint* is a unique signature of a page that is saved in the system and used to automatically classify incoming documents against those documents that have been processed before. The idea is that, if the fingerprint of the incoming document matches an existing one, you can safely assume that the incoming document is of the same class as the existing one. This technique is particularly well adapted for structured and semistructured documents, such as forms, that exhibit a fairly constant layout.

A fingerprint is made up of a sample image with a representative layout of the class of document and information that describes its geometric profile based on analyzing its pixel distribution. It can also be complemented with recognition results. The fingerprint is assigned a unique identifier that is saved in the fingerprint database.

Although all the documents of a same class look alike from a distance, every instance of a document is typically unique in that its actual contents are different from the others. Therefore, the chances of detecting the exact document are highest when the incoming document is a copy of the original document. Detecting an instance of an already identified class is a matter of measuring the proximity of the fingerprint of the incoming documents to the existing ones. The closest match has the highest probability of the instance belonging to the identified document class.

Fingerprints get perfected over time with additional information when more instances of the same class are routed to be identified by operators in the Datacap process. By using the fingerprinting and *Intellocate* libraries, you can configure your application to automatically create a new fingerprint and save zone positions after an unrecognized page has been routed to a Verify task. This way, only verified recognition results are saved in the fingerprint, enhancing accuracy.

Bar code-based identification

In bar code-based classification the type of page is associated with a given bar code value that is recognized on the image.

To configure this classification method you simply need to specify the 1D or 2D bar code type, its orientation on the page, the expected value, a minimum confidence level, and the type of page it maps to. When the page identification ruleset is run on the batch, each page is then tested against the collection of bar code-page type mappings, and when the expected value is detected the corresponding page type is associated with the image. If you have the option of designing your own documents, this is one of the most reliable and efficient methods to classy documents.

Structure-based identification

Structure-based identification is used in cases when the batch is fairly structured, that is, when the succession of pages can be predicted and used to determine the document type. In such case, you can use the actions of the runtime Document Hierarchy to arbitrarily set the page types.

Text and pattern matching

Text and pattern matching techniques are used on structured and semistructured documents. They are used when the types of documents are so close and the relative positions of zones so constantly changing that fingerprinting is unable to detect the type accurately.

Rather than looking at the page, as fingerprinting does, text matching attempts to identify a document based on keyword and phrase searches that unequivocally determine the type of document. It is called after the document has run through recognition, and therefore, typically requires more processing than just image analysis. Pattern matching concentrates on specific graphical marks or anchors in areas of an image.

Identification with IBM Content Classification

The techniques mentioned previously all look for specific features of the document to identify and separate it from others in the batch. However, they work less well in cases when you need to process a mix of mostly text documents. Examples include miscellaneous customer correspondence, complaint letters, policies, statements, or affidavits with no predictable structure, logo, bar code, marks, or keywords. In such cases, you must understand the content in the same way as a person who, unable to recognize a type of document at first glance, needs to read its content to identify the document. For this reason, Datacap relies on IBM Content Classification.

Important: Depending on your licensing model, IBM Content Classification might require additional licensing from IBM.

Similar to fingerprinting, IBM Content Classification creates a unique identity exclusively from the textual contents of documents. It looks for patterns, concepts, and associations and stores the results mathematically. This identity is then associated with a given type of document. Initially, document identification requires human intervention to match a given identity to a document type. However, IBM Content Classification can learn from the processing of a range of sample documents, and over time it requires no manual intervention.

At run time, the IBM Content Classification connector invokes IBM Content Classification and passes to it full-text recognition results. IBM Content Classification analyzes the content and compares it to its collections of identified types of documents. If it finds a match, it returns the type to Datacap. Otherwise, Datacap assigns a low confidence rating to the document, which causes it to be classified by an operator.

Because IBM Content Classification analyzes documents in their entirety based on concepts, it has a much larger scope to accurately identify a document than other methods can do, based strictly on a linguistic approach. The internal representation of information in IBM Content Classification also makes it less sensitive to OCR or ICR and manual input errors.

In summary, IBM Content Classification:

- Automatically identifies text-intensive, free-form documents
- Reduces prescan manual sorting and document separating
- Enables automatic processing of mixed document batches
- Processes noisy OCR or ICR documents without operator intervention

Manual page identification

Manual page identification is used as a last resort when every other method has failed. With the help of automatic fingerprinting or content-based classification with IBM Content Classification, the number of manual interventions should decrease over time.

2.3.6 Recognition and text manipulation

Recognition is concerned with the process of converting the contents locked in the images of paper or electronic documents to live data that can be put to use, such as indexing documents and driving the business workflow, producing full-text searchable documents for analytics, or feeding data to line-of-business systems.

In addition to recognition, Datacap provides several capabilities to locate, validate, manipulate, and look up extracted data in support of classification, redaction, and quality assurance.

Bar code recognition

The bar code detection capabilities are used to automatically locate and recognize one or several bar codes in an image. Bar codes are used to store data that can be detected with high accuracy, even on poor quality documents.

Many types of bar codes have been developed over the years to serve the needs of specific industries, such as retail, logistics, manufacturing, postal services, healthcare, airlines, and transportation in general. The choice of using a specific bar code in a Datacap application can be dictated by the bar code standard in use by the customer. Alternatively, you can decide on the choice of bar code arbitrarily to satisfy the internal needs of the application. Such needs might be to identify a type of form, mark document boundaries in a batch, or reconcile incoming documents with a work item or other documents that belong together.

Datacap can detect 1D and 2D bar codes. One-dimensional bar codes (Figure 2-6) store a limited amount of data, typically a single alphanumeric string that can be used as a reference. They are coded by using a pattern of vertical lines of varying width read along the horizontal axis of the bar code.

Figure 2-6 One-dimensional bar code Code 39

For example, the Code 39 bar code can be used to code a claim number on outgoing correspondence to a customer to automate the claim identification process when the mail returns. Code 39 can store up to 43 alphanumeric characters.

Two-dimensional bar codes can store up to several kilobytes of data. They are coded by using a matrix that represents information along the vertical and horizontal axes of the bar code.

The PDF417 bar code is a popular two-dimensional bar code (see Figure 2-7) that has many uses including to code driver license information in certain states. It codes information in multiple rows (from 3 to 90) that show clusters of bars and spaces. It is described as a "portable data file," because the bar code carries the information, not just a reference to it.

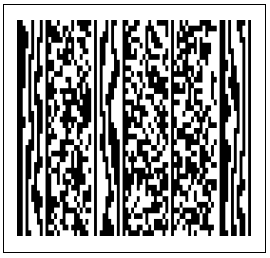


Figure 2-7 Two-dimensional bar code PDF417

For example, this PDF417 bar code encodes the entire postal address: "International Business Machines, 3565 Harbor Boulevard, Costa Mesa, California, 92626-1405, United States."

A full list of the 1D and 2D bar codes supported by Datacap can be found in the Datacap documentation.

A special type of bar code is the patch code (Figure 2-8), which is typically used on sheets of paper that are inserted in a batch to separate documents. Unlike a bar code, a patch code needs to be positioned precisely parallel to the leading edge of the page in the scanner transport.

Figure 2-8 Patch code type 2

A patch code consists of a pattern of 4 horizontal bars with 2 levels of thickness which results in 6 combinations. Each patch code type has a specific use. For example, type 2 indicates the start of a new document, whereas type 4, also called the toggle patch code, can be used to indicate the start and end of a portion of the batch with color images.

Optical character recognition (OCR)

OCR technology is used to convert machine-printed text in an image to editable text. It is at the core of the Datacap document identification and data capture process and is used for full page recognition and zonal data extraction. Datacap includes two OCR engines that can be used individually or in combination to offer the best possible results.

Although implementations differ, the two engines operate in a similar fashion. At a high level, the OCR engine analyzes the image and the textual zones and isolates individual characters. It then compares each character to a collection of template character bitmaps (in various fonts) and selects the closest match. It assigns each character a "recognition confidence level" based on how well it correlates with the template. Then it assembles the characters into words and resolves ambiguities by using dictionaries or lexicons and various other techniques.

The confidence levels are measured against the confidence thresholds set in Datacap. The higher the OCR confidence threshold is, the higher the number of errors. Confidence levels of recognized zones are saved in the Document Hierarchy at run time to be used to color code the fields and drive the focus to problem fields in the Datacap Verify user interface. It is also possible to run multiple OCR engines on the same image to achieve the most accurate results across them all, which is a technique called *voting*. In that case, in each successive pass, Datacap compares the confidence level of the recognized data in the current pass with the one recorded in the previous pass. It then only updates the field with the new data and the confidence level if it is higher.

Although the accuracy of OCR engines has improved over the years, it is affected by many factors. These factors include the page layout and background, image resolution, color/contrast/brightness, skewing, jagged characters, font type, size, and emphasis, type of compression, language, and character set. However, in most Datacap implementations, the following factors can be adjusted to produce the best possible results:

- Resolution, color, brightness, contrast, and skewing can be tuned at the scanner level or corrected by using Datacap image cleanup and enhancement actions.
- Character attributes, such as jagging, thickness, or thinness, can also be corrected by using image enhancement actions.
- Page layout, font type, and font size can be adjusted if the organization can influence the design of forms.
- Uncompressed or CITT Group 3/4 TIFF can yield better recognition results. In such case, Datacap offers the ability to convert incoming images with other formats and compression schemes to TIFF. They can still be carried through the Datacap workflow by using a separate stream to the repository. Alternatively, Datacap can convert the TIFF images to a more compressed format at the Export stage.
- Recognition performance of national languages and character sets (especially accented characters) can be adjusted by selecting the OCR engine that yields the best results or by using voting.

Intelligent character recognition (ICR)

ICR technology converts hand-written characters in an image to editable text. The overall operating principle of ICR is similar to OCR. However, because of the variability in handwriting, the techniques to separate and classify characters are different. Rather than trying to isolate and match whole characters, individual handwriting strokes are isolated and analyzed spatially to see which strokes most likely belong to which characters.

Also, character classification requires a much broader base of shapes and more complex methods, including statistical probabilities, to eliminate ambiguity and identify characters and words with confidence.

Figure 2-9 shows an example of ICR on a tax form.

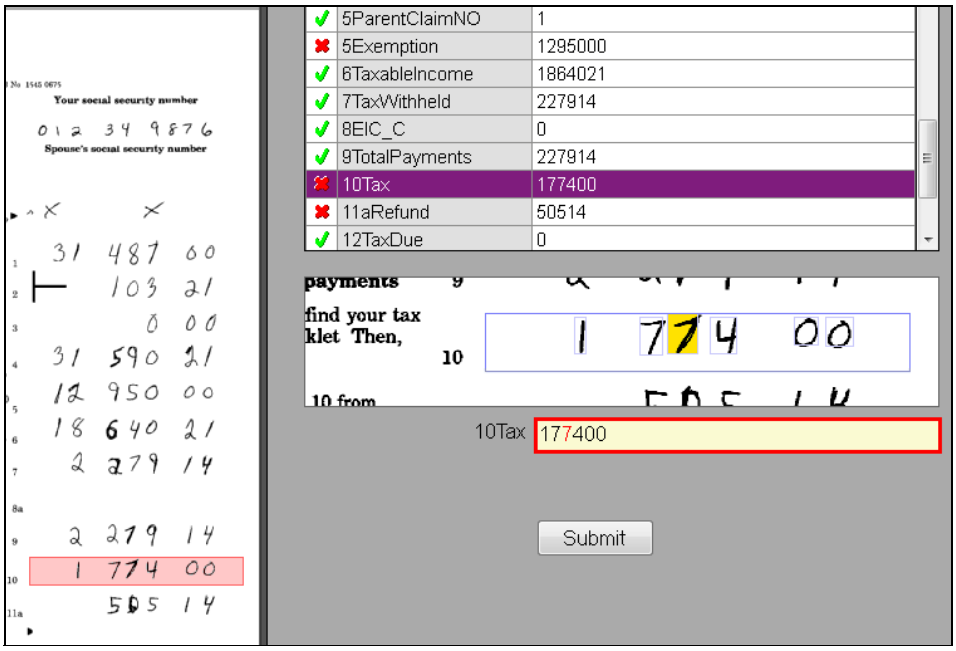


Figure 2-9 ICR on tax form

ICR is affected by the same factors as OCR. However, it is affected by the clutter introduced by handwriting going over the boxes used in form fields and for normalizing character spacing (“constraint boxes”). Line removal or repair and dropout are typically employed to make the text stand out and improve accuracy.

Optical mark recognition (OMR)

OMR is used in Datacap to detect whether check boxes, bubbles, or other types of marks have been selected. It is typically used in combination with other Datacap functions that applies processing logic to the basic OMR results to interpret and turn them into actionable data. The results depend on the purpose and type of prompt or answer expected. For example, the answer might be yes or no, yes or no to all that apply, multiple choice, grid to form numbers or words, or add up marks.

Although this technology has been in use for years, it is not easy to address the many situations that can arise and to determine the confidence levels and errors that trigger the Verify task. Factors that affect accuracy include the type of marks, filling method, variability in the response of the filler (spill over, too light, erasure), and interfering specks and background noise in the image. See Figure 2-10.

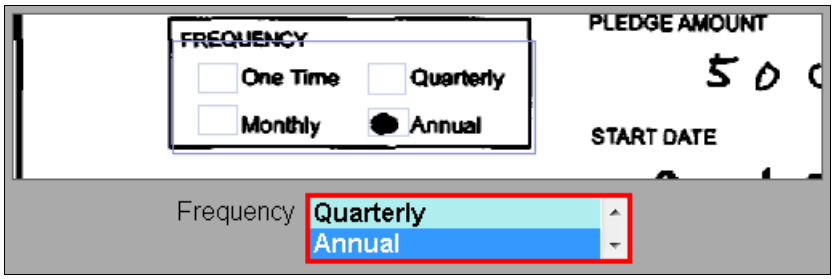


Figure 2-10 OMR used in a survey application

Datacap provides flexibility to address these cases. When configuring the zones and fields, you define a parent field (for example, “Frequency”) as an OMR group and subfields (for example, “One Time,” “Monthly,” “Quarterly,” and “Annual”) to host options that belong together. You also specify in the parent field the number of options and whether multiple ones can be selected.

To do the actual recognition, Datacap offers two methods to address various situations. In most cases, when regular check boxes are used, the quickest setup is to use the OCR_A (ABBYY) engine. When the outline of the check box is removed by line removal or dropout, which is sometimes necessary to process high-density forms, or when using bubbles or unusual check marks, you might need to use the “pixel evaluation method” instead. In this method, the zone defined for the check box is evaluated against a threshold of darkness (black pixel percentage) and a threshold of background noise that you specify to determine what is considered a checked mark. The difficulty with this technique is that it is more sensitive to variations in background noise or specks, which affects the pixel count for the zone. Therefore, adjustments might be necessary to find the appropriate values of these thresholds.

Locating text

Datacap offers an extensive library of actions that can be used to search the recognized text for specific words and regular expressions and to navigate around a page based on the positions of lines and words detected by the OCR engines.

These actions are used in instances when text location cannot be predicted, such as in semi-structured or free forms, but when specific keywords or text can be expected in labels and headers. They include the capability to form lists of keywords, such as `Invoice`, `InvNum`, `Invoice #`, and regular expressions, such as the one that follows, for a US Social Security number, and to iterate through them until matches are found:

```
[\^\b\s\n\r][0-9]{3}[\-]{1}[0-9]{2}[\-]{1}[0-9]{4}[\b\s]*
```

When a particular textual form is located, you can use navigation actions such as moving up/down several lines, or right or left several words, to find and manipulate the target data with other actions, such as those used for validation or to process line items and tables in purchase orders, invoices, or shipping manifests.

Combining recognition techniques for successful extraction

Successful content extraction depends on combining three aspects of recognition techniques:

- The correct recognition engine on the correct type of content and language
- Judiciously defining the recognition scope, that is, zonal versus full page recognition, depending on the type of page layout
- The method used in the application to target the text to be extracted, depending on the variability of the page layout

We have seen in the previous sections that OCR is designed for machine print and ICR for hand-printed characters. Recognition accuracy, however, also depends on the national language being recognized, especially when character sets are different from one language to another, such as Russian and English. Within the same character set, accuracy can also be improved by precisely selecting the locale, because specific dictionaries can be used to disambiguate wrongly recognized words. Also, some recognition engines can be better than others on certain fonts or languages, so it is possible to improve accuracy by using voting to run different OCR engines on the same document and to keep the best result of the combined passes.

For information about the languages supported by Datacap, see the Datacap Language support page on the Web:

http://www.ibm.com/support/docview.wss?uid=swg27044111

If the full content of a page is not required, zonal recognition typically performs much faster than full-page recognition and works both for machine and hand-printed characters. However, zone positions need to be reliable across documents of a given type. They are stored in Datacap as part of the fingerprint to delineate the parts of the image where the recognition needs to take place. If the page layout changes, the recognition engine will look at the wrong place and produce unreliable results.

When layout variability is too great, a better strategy is to use full-page recognition and search the extracted text content for the target strings or search by using regular expressions. When the OCR engine processes a full page, it analyses the image to find and identify the zones of interest, such as the blocks of text, text lines within the blocks, words within the lines, photo areas, tables, and so on. The engine stores their positions together with the recognized content so that it can be used later by Datacap to navigate the text from the page.

For example, a medical record number in a block of text that flows with the preceding text can be extracted by searching for the string `Medical Record #` and then navigating immediately to the right and capturing the actual number, as shown in Figure 2-11. Similarly, a Social Security Number (SSN) can be directly extracted from the content by using a regular expression that filters an SSN number with 3 numbers, dash, 2 numbers, dash, and 4 numbers.

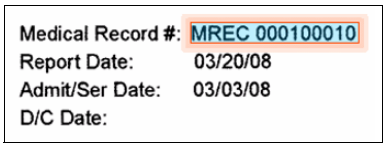


Figure 2-11 Navigating the recognized text to extract data

Validation

Datacap offers an extensive library of elemental actions to validate and manipulate data captured in the objects of the runtime Document Hierarchy and to ensure that they conform to your business rules.

The following actions are possible:

- Data formatting actions to normalize field values or prepare them for calculation or comparison purposes, including the following actions:
  - Padding with zeros or spaces to match the expected number of characters
  - Deleting a specific character in a specific position or all instances
  - Deleting a class of characters from a field (alpha, numeric, punctuation, non-alpha numeric, or system characters)
  - Testing data types (date, currency, alpha, or numeric) and field length
  - Converting the case of characters or value to currency
  - Clearing a field value
  - Inserting a decimal point or a specified character in an existing field value
  - Trimming spaces and truncating and splitting field values
  - Parsing postal addresses and names and populating individual fields

- Manipulation of field values and document hierarchy variables, including the following actions:
  - Assigning default values to fields
  - Copying or appending values between fields
  - Comparing field values and verifying arithmetic calculations between fields
  - Comparing dates and testing them within a range or days
  - Assigning data or a time stamp to a field
  - Testing field content: Filled, empty, max or min. length, specified matching value, or percentage numeric or non-numeric data
  - Testing the number of OMR boxes and whether they are selected
  - Testing a regular expression in a field
  - Testing variables and assigning variables to fields
  - Summing up values of subfields
- Invoking a message box to provide guidance in the Verify task

  For example, by using combinations of these actions, you can create rules and attach them to any object of the Document Hierarchy to test the following information:
  - Values are within accepted ranges
  - Data is formatted as required
  - Dates are valid and deadlines are met
  - Numbers add correctly or are not missing
  - Mandatory fields and check boxes have been completed
  - Dependencies between data are respected
  - Data matches sets of permitted values, as checked by using lookup actions

  Upon failure of any of these rules, Datacap flags the associated field and page for manual review in a Verify task, similar to actions for low-confidence recognition results.

Lookups

As a good practice, check and normalize data early in the business process to reduce errors and enforce consistency. To achieve this objective, Datacap provides a set of actions that you can use to connect through ODBC to a business database hosting reference information to run SQL queries to populate fields in the runtime Document Hierarchy. Such information might include customer names, part numbers, and geographical information, such as states and locations of branch offices.

On the client side, Datacap Navigator also offers the ability to use the External Data Service (EDS) capability of IBM Content Navigator for lookups. When an external data service is configured for a certain action or property, the service is invoked automatically every time a user interacts with that item in Datacap Navigator.

2.4 Delivering documents and exporting data

The Datacap workflow completes the processing of a batch by persisting the captured documents and metadata to an enterprise content management (ECM) repository. In many cases, it also exports some of the business data to line-of-business systems such databases or applications. It can output data and contents in a format that is compatible with the input stage of another system, such as IBM Content Manager On Demand.

Datacap includes libraries to export to IBM and non-IBM ECM repositories, relational databases, XML, and flat files. Datacap integrates with IBM FileNet Image Services, FileNet Content Manager and IBM Content Manager by using their respective native APIs. Datacap can also communicate with these or non-IBM repositories using the Content Management Interoperability Services (CMIS) interface. To use CMIS with an IBM content repository, the services must be enabled for that repository.

The CMIS interface can be used both for importing document definitions in the Datacap Document Hierarchy and for exporting the documents and extracted data to any CMIS-enabled repository, including the cloud-based IBM Navigator.

Note: For more information about Datacap's export capabilities, see Chapter 9, "Export and integration" on page 205.

2.4.1 Mapping repository document properties using CMIS

Datacap field definitions used in the Document Hierarchy, including name, type and length, can be imported and synced with the property definitions of any CMIS-enabled repository, including on-premises IBM ECM repositories.

For example, you can get access to the document classes of an IBM FileNet Content Manager repository by creating a Datacap application type of *CMIS*, using the Application Wizard in FastDoc or Datacap Studio. The FileNet Content Manager property definitions can be attached to any level of the Datacap Document Hierarchy, not just to the document level. This way, batch-level fields in the Document Hierarchy that hold data relevant to all the documents included in the batch are then exported to each of the FileNet Content Manager documents. See Figure 2-12 for an example.

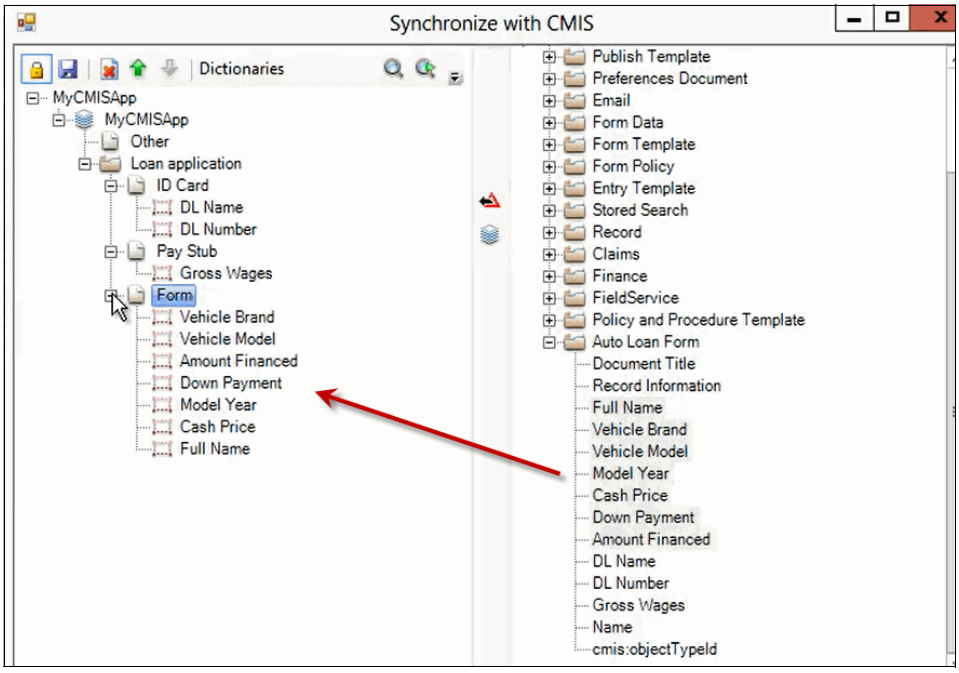


Figure 2-12 Mapping of FileNet Content Manager document properties to Datacap fields

2.4.2 Exporting to FileNet Content Manager

Datacap provides the Export to FileNet Content Manager compiled ruleset to write documents to FileNet Content Manager. It displays a user interface to configure the following capabilities:

- Establish a connection to a FileNet Content Manager system.
- Attach to a given object store and FileNet Content Manager document class.
- Define a root folder and create a subfolder to store documents. For example, you can create a new folder for each new claim based on a unique claim identification number extracted from the claim form.
- Assign the FileNet Content Manager document class to export to and map the Datacap field values and variables of the runtime Document Hierarchy to its properties to index the documents in the repository. When using imported property definitions the mapping is already provided. Note that you can assign any MIME type to the documents; not just image types. This enables you to store all sorts of electronic documents (PDF, Excel, and so on) if needed.
- Upload documents to the destination folder.

Depending on the use case, the export rulesets can be preceded by a document conversion ruleset in which the individual pages of the Datacap document are merged into a multipage TIFF or PDF file. Alternatively, each Datacap page can be uploaded to FileNet Content Manager as a separate file according to “content element.” This enables the retrieval of individual pages, on demand, from FileNet Content Manager applications to conserve bandwidth and to provide a better response time.

2.4.3 Exporting to IBM Content Manager

Datacap provides the “Export to IBM Content Manager” compiled ruleset to write documents to IBM Content Manager. It displays a user interface to configure the following capabilities:

- Establish a connection to the IBM Content Manager system.
- Set the destination folder to store the documents. For example, you can set the destination folder to upload a new claim form to an existing customer’s folder, which you assign by using data extracted from the scanned document.
- Create and index (assign property values to) a folder of a specified type. The folder can be attached to a parent folder to create a folder structure. For example, you can create a new folder for each new claim, index the claim folder with a unique claim number extracted from a unique bar code on the scanned claim document, and attach it under a folder named after the customer’s unique ID, which is also found on the claim document.
- Assign the IBM Content Manager document item type to export to and map the Datacap field values and variables of the runtime Document Hierarchy to its properties to index the documents in the repository. When using imported property definitions, the mapping is already provided. You can assign any MIME type to the document, not just image types. This enables you to store all sorts of electronic documents (PDF, Microsoft Excel, and so on) if needed.
- Upload documents to the destination folder.

The integration with IBM Content Manager does more than just delivering content into a static foddering folder structure. The Datacap actions provide significant flexibility in the way documents are organized and stored in the repository, including the following capabilities:

- Search for an existing folder or document in the repository that matches a specified attribute and value. For example, you can configure the export to search for a specific folder using a unique customer ID, or for a specific document using a claim number extracted from the scanned documents, and then use the folder or document reference returned by IBM Content Manager for further actions.
- Add to, delete from, or replace pages of an existing IBM Content Manager document. For example, you can search for a specific claim document based on a unique claim number extracted from the scanned Datacap document and add all or specific pages from it to the IBM Content Manager document. Alternatively, you can delete some or all from the IBM Content Manager document, or you can simply replace a given page by another from the Datacap document.
- Create a child component (multiple values, or *multi-valued* property) under the current document and assign attributes to it. For example you can add sets of multiple-value fields to a claim document to store the information about each car of a multiple-collision accident. For each line item that describes a damaged car in the police report, you can create a child component called “damaged vehicle” with three attribute-value pairs for the VIN, Driver First Name, and Driver Last Name.

2.4.4 Exporting to a CMIS repository

Datacap provides the CMISClient actions library to export documents to an IBM Content Management Interoperability Services (CMIS) repository. The CMIS export is the preferred method to commit documents to a remote repository over the Internet. It provides the following capabilities:

- Establish a connection to the CMIS-enabled repository.
- Create a folder in the root or a parent folder. This is useful for example to create a new folder, based on the claim number extracted from a claim form, to store all documents pertaining to the claim.
- Assign a document type and map the Datacap field values and variables of the runtime Document Hierarchy to its properties to index the documents in the repository.
- Upload the content file associated with the document type and properties created earlier. If each Datacap document contains multiple pages, the CMIS export actions must be preceded by a document conversion ruleset to convert the individual pages of the Datacap document into a multipage TIFF or PDF file. You can assign any MIME type to the document, not just image types. This enables you to store various kinds of electronic documents (PDF, Excel, and so on) if needed.
- Test for the existence of a file or folder. For example, before creating a new folder for a claim, this action can be used to test for the existence of a folder named after the claim under the same parent folder and create it only if it does not already exist. If it exists, you just add the new documents to the existing folder.
- Delete a folder or file. This is useful for repositories that do not implement a versioning mechanism to publish the latest version of a document. For example, you can test for the existence of a policy document and then delete and replace it by the newly uploaded one.

Note: A folder cannot be deleted until all the files in it are deleted.

Figure 2-13 shows and example of a CMIS export ruleset.

```
Export to CMIS
  Batch
    Function1
      CMISLogin ("https://cmis.foo.com:9999/fncmis/resources/Service", "cmisUser", "28ED274F30B1420785B0F1FDE", "OS26987")
  Page Close Claim_Form
    Function1
      CMISSetVersion ("MAJOR")
      CMISSetDocUploadType ("AC_ClaimForm")
      CMISSetDocUploadProperty ("AC_PolicyNumber", "@P\Policy Number", "string", False)
      CMISSetDocUploadProperty ("AC_DriverName", "@P\Driver Name", "string", False)
      CMISSetDocUploadProperty ("AC_VehicleLicense", "@P\Vehicle License", "string", False)
      CMISSetDocUploadProperty ("AC_VehicleColor", "@P\Vehicle Color", "string", False)
      CMISSetDocUploadProperty ("AC_ClaimNumber", "@P\Claim Number", "string", False)
      CMISSetDocUploadProperty ("AC_IncidentNumber", "@P\Incident Number", "string", False)
      CMISSetDocUploadProperty ("AC_PolicyFullName", "@P\Policy Full Name", "string", False)
      CMISSetDocUploadProperty ("AC_FormID", "@P\Form ID", "string", False)
      CMISSetDocUploadProperty ("AC_VehicleVIN", "@P\Vehicle VIN", "string", False)
      CMISSetDocUploadProperty ("AC_IncidentTime", "@P\Incident Time", "string", False)
      CMISSetDocUploadProperty ("AC_IncidentDate", "@P\Incident Date", "datetime", False)
      CMISUploadPage ("@BatchID", "/Insurance/Auto Claims", "image/tiff")
```

Figure 2-13 Actions to export to CMIS in Datacap Studio

2.4.5 Exporting to a database

To export to a database that is accessible through ODBC, the ExportDB library provides the following actions:

- Establish and close a connection to the database.
- Open the target database table.
- Assemble each database record in memory, and populate it with data from Datacap field values and variables of the runtime Document Hierarchy.
- Commit the database record.

You bind the actions to open the database to the open event at batch level. You also bind the actions to close the database to the *Close* event at batch level. In addition, you bind the actions to create the data records for the Open event at document level. See Figure 2-14 on page 57 for an example.

```
ExportDB
  Rule: 1040EZ
    Function1
      ExportOpenConnection ("@APPVAR(*/exportdb:cs)")
  Rule: Page_1040ez
    Function1
      SetTableName ("Data")
      ExportBatchIDToColumn ("db_BatchID")
      ExportFieldToColumn ("TaxpayerName,db_TaxpayerName")
      ExportFieldToColumn ("SpouseName,db_SpouseName")
      ExportFieldToColumn ("Address,db_Address")
      ExportFieldToColumn ("City,db_City")
      ExportFieldToColumn ("State,db_State")
      ExportFieldToColumn ("Zip,db_Zip")
      ExportFieldToColumn ("TaxpayerSSN,db_TaxpayerSSN")
      ExportFieldToColumn ("SpouseSSN,db_SpouseSSN")
      ExportFieldToColumn ("1TotalWages,db_TotalWages")
      ExportFieldToColumn ("2TaxableInterest,db_TaxableInterest")
      ExportFieldToColumn ("3Unemployment,db_Unemployment")
      ExportFieldToColumn ("4AdjustedGross,db_AdjustedGross")
      ExportFieldToColumn ("5Exemption,db_Exemption")
      ExportFieldToColumn ("6TaxableIncome,db_TaxableIncome")
      ExportFieldToColumn ("7TaxWithheld,db_TaxWithheld")
      ExportFieldToColumn ("8EIC_C,db_EIC_C")
      ExportFieldToColumn ("9TotalPayments,db_TotalPayments")
      ExportFieldToColumn ("10Tax,db_Tax")
      ExportFieldToColumn ("11aRefund,db_Refund")
      ExportFieldToColumn ("12TaxDue,db_TaxDue")
      ExportFieldToColumn ("TaxpayerSignature,db_TaxpayerSignature")
      ExportFieldToColumn ("SpouseSignature,db_SpouseSignature")
      AddRecord ()
  Rule: 1040EZ_Close
    Function1
      ExportCloseConnection ()
```

Figure 2-14 Actions to export to a database in Datacap Studio

2.4.6 Exporting to a flat file

By using the Datacap Export library, you can output data to the file system to be picked up for import by another system. The library provides the following actions:

- Set the path, file name, and extension of the export file.
- Format text (new line, blank lines, fields, filler characters, value and OMR separators, value justification, value length, OMR separator, and so on).
- Output text, date, time, field values, and variables of the runtime Document Hierarchy, and filter on field status.
- Save the export file.

The information that needs to be exported and how to code the export file depends on the target system. For example, this method can be used to feed the FileNet Content Manager Bulk Import Tool as an offline alternative to a direct connection to FileNet Content Manager. Alternatively, it can be used to import documents in IBM Content Manager On Demand by using its ARSLOAD utility.

You bind the actions to create the export file and write the data needed once to the *Open* event at batch level. Then, you bind the actions to output the information required to the Open event at document level. Then, you bind the action to save the file to the *Close* event of the batch.

2.5 Datacap user interfaces

Datacap offers several user interfaces to cover the needs of users, administrators, and superusers who configure applications. This section focuses on the following user interfaces:

Datacap Navigator A rich web client that is based on the IBM Content Navigator technology that is used in distributed imaging operations.

Datacap Desktop A thick Microsoft Windows-based client, well-suited for high-volume production imaging scenarios such as centralized mailroom operations.

Datacap FastDoc A versatile thick Windows-based client that can function both as a Datacap client and as an application configurator for business superusers.

Datacap Mobile An iOS or Android app that can be used either stand-alone or integrated into a custom app as part of the business transactions you offer to your customers.

2.5.1 Interface productivity features

Datacap provides flexible user interfaces with innovative capabilities to enhance the productivity and experience of an operator.

In all thick and web user interfaces, Datacap provides hot keys and visual cues to makes it easy to quickly focus attention on and navigate to problem areas. It provides the capability to position image snippets next to the corresponding recognized data fields. This way, operators can easily compare recognition results to the zones where the data originated.

You can configure color-coded backgrounds and character ink to indicate the confidence levels on recognized data and quickly show validation errors. For example, as Figure 2-15 shows, you can use the following backgrounds:

- Blue background for high-confidence recognition results and no data validation errors
- Yellow background for low-confidence recognition results with red ink for low-confidence characters in the field
- Red background for data validation errors

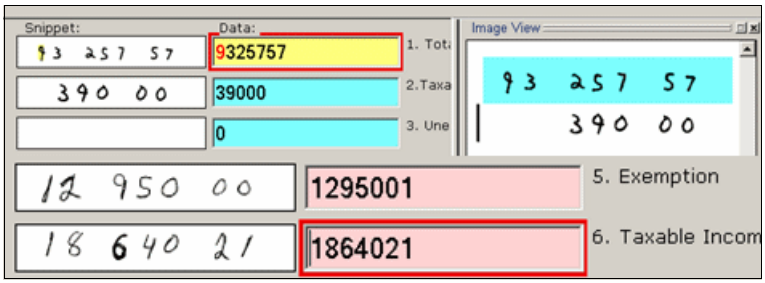


Figure 2-15 Datacap image snippets and color-coded user interfaces

With the *Click N Key* capability, you can pull data into a field with a single click. You position the cursor in the destination field and then click or draw a box around the piece of information in the image that you want to grab. See the example shown in Figure 2-16.

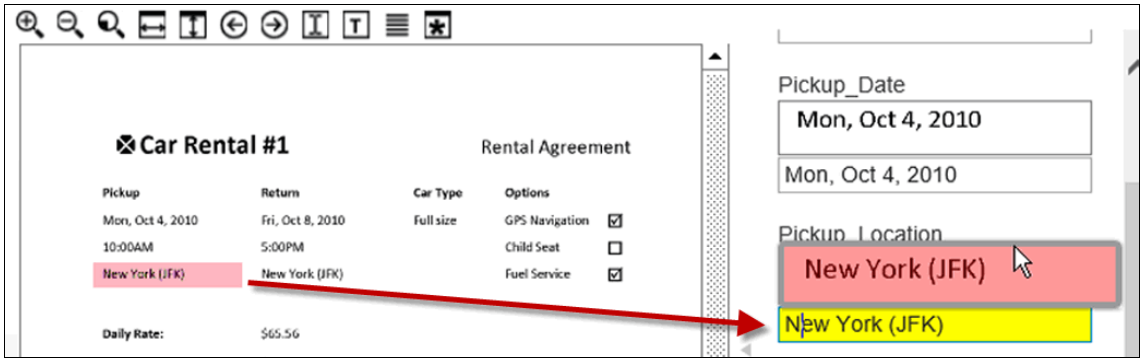


Figure 2-16 Click N Key capability in Datacap

You can also combine this feature with Datacap's ability to learn from the an operator's manual input to record the location where the data was found in the fingerprint that is associated with the type of document. The new information learned by Datacap is used to fine-tune and automate the recognition process the next time it processes a similar document.

2.5.2 Datacap Navigator

Datacap Navigator is a web user interface based on state-of-the-art IBM Content Navigator technology, which provides a rich and responsive user experience in a familiar environment, consistent with the other products in the IBM ECM family. If authorized, the user can also access ECM repositories or any other functions they are allowed to access.

It is built on HTML 5, JavaScript and Dodo technology, which enables Datacap to interoperate with other ECM applications such as Case Manager that support the same infrastructure.

It provides interfaces for users to scan, classify, and verify documents, and for supervisors to monitor the jobs being run on the system. It also provides the capabilities for administrators to configure the processing flow of applications, create Datacap users and assign functional security, and so on.

Note: For more in-depth information about Datacap Navigator, see Chapter 10, "Datacap user experience in IBM Content Navigator" on page 233.

2.5.3 Datacap Desktop

Datacap Desktop is a thick client designed for high throughput, and is typically run by scan operators using customized and optimized user interfaces. It is used to scan or import images, classify documents and fix batch structure issues, and verify captured data.

The Datacap Desktop is made up of three main functions:

- Application selection and the Task List to run the tasks that have been defined for an application
- The scan and batch fix-up user interfaces that connect to a scanner or provide access to the file system for file import

- The verification user interface to review extracted data and resolve validation or low-confidence recognition problems

Task list and execution

You log in to the Datacap server and select among the applications you have been given access to. Depending on the permission settings of the application, you are automatically presented with the next pending task or with a Task List (Figure 2-17) that displays the jobs of the application at various stages of execution, allowing you to pick.

When selecting a job, the viewer displays detailed information about the batch and all the pages it contains. You can also filter and sort the jobs on any column, reorder the columns through a simple drag action, and select a specific set of columns, including custom columns that have been added to the batch table. The list can be paginated to a fixed number of items to maintain performance when a large number of jobs are returned.

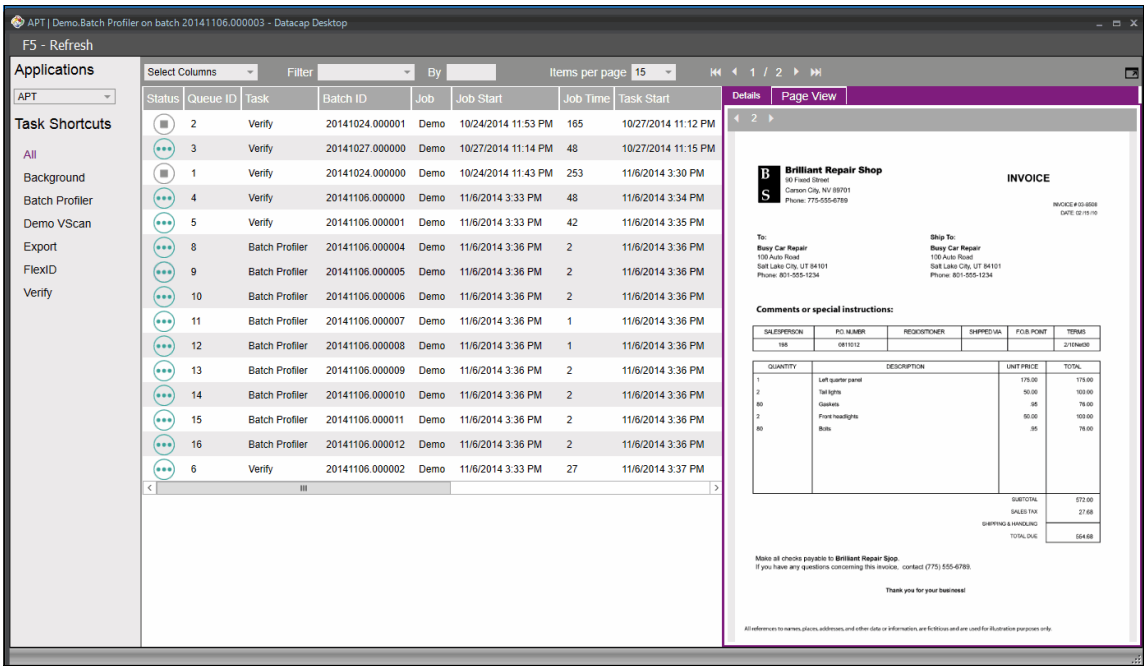


Figure 2-17 Datacap Desktop Task List

When selecting a job, you can start the next pending task associated with it or directly run a specific task by using the Task Shortcuts list that appears on the left side, under each application. A job typically starts with a scan or import task and progresses according to the Datacap workflow defined for the application, with “foreground” (user) tasks, such as Classify or Fix-up, presented to the user and background tasks, such as clean-up and classification, automatically run by the Rulerunner process. However, when developing and testing your application, you can stop the Rulerunner and have the Datacap Desktop run one task at a time, manually, by simply selecting and double-clicking the job in the Task List.

Scan and batch fix-up

One of the primary functions of Datacap is scanning documents. Datacap Desktop supports the operation of scanners that use industry-standard TWAIN and the proprietary ISIS drivers.

The scan operator is able to scan in pages and make adjustments to the batch, such as reordering, removing, or re-scanning pages, ensuring the batch has the correct structure as it goes to the next step in the workflow.

Figure 2-18 shows the default scan interface in Datacap Desktop.

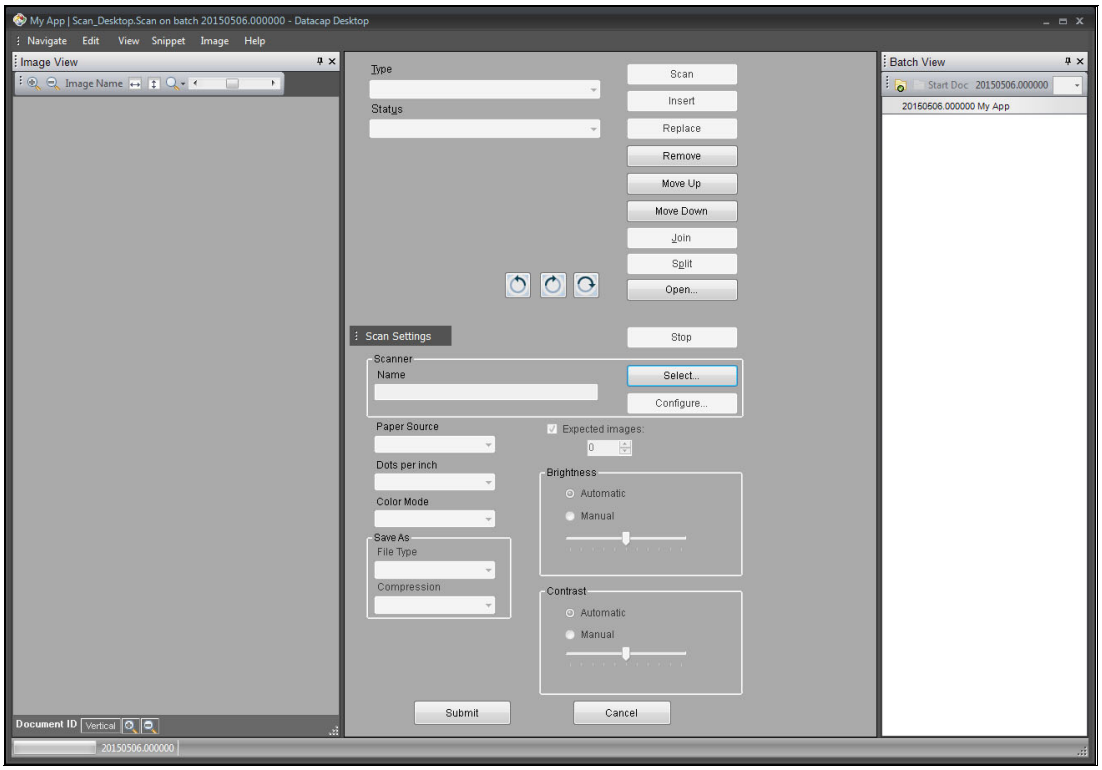


Figure 2-18 Datacap Desktop Scan Interface

In cases where the documents already exist in electronic form (on the file system, for example), Datacap Desktop provides the ability to “scan” them from disk. This is also useful when developing and testing your application. Repeatedly scanning the same documents is impractical and might prevent you from pinpointing and resolving issues in your application as you work on it. See Chapter 5, “Designing a Datacap solution” on page 113 for more information about designing and developing Datacap solutions.

The scan operator’s help is again required if one or more pages fail the page identification step, that is, if Datacap is unable to automatically identify one or more of the documents and pages that comprise the batch. When this happens, the batch can be routed to a Fix-up task, and the operator can add, remove, or merge documents and pages, as shown in Figure 2-19 on page 62. The operator can even scan in additional pages if necessary.

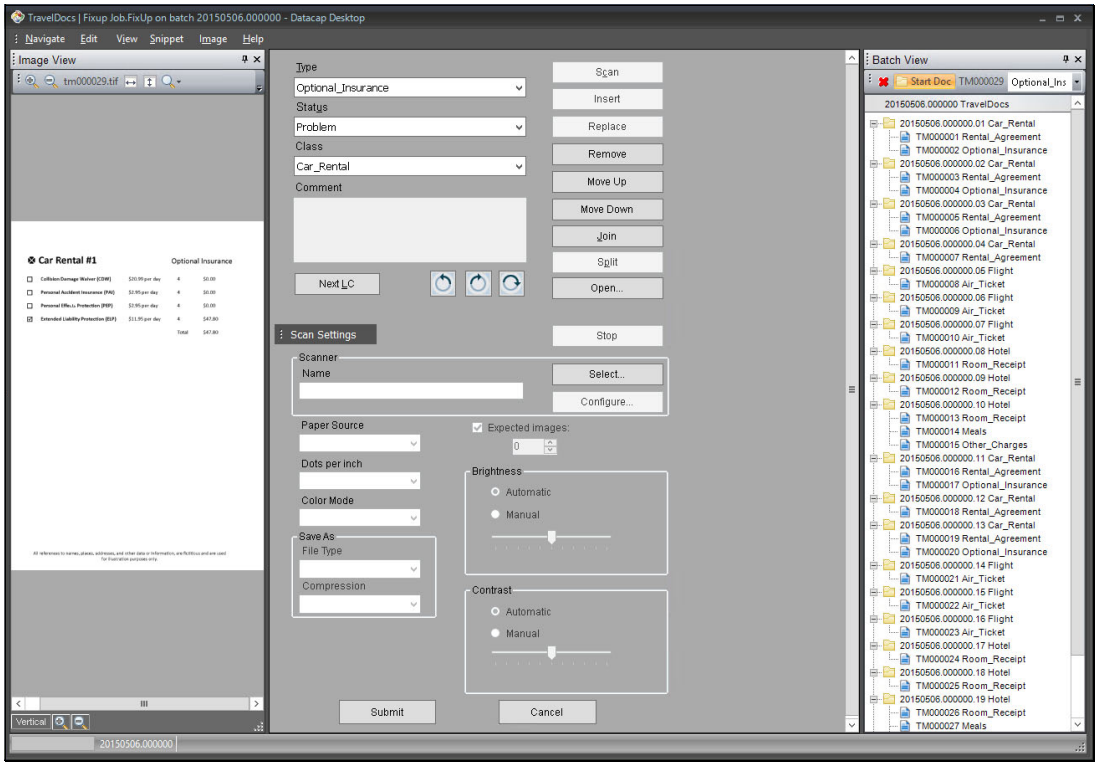


Figure 2-19 Datacap Desktop Fix-up Interface

Verification

If a field has one or more low-confidence characters or if it fails a business rule (for example, a value might only be numeric but a letter has been found by the OCR engine), the operator is prompted to make the necessary corrections before automatic processing resumes. If a page has no problems, the system can be configured to skip it so that the user only reviews pages that need attention.

The system color codes the fields. The fields that are low-confidence are displayed in yellow, and fields that have validation errors are displayed in red. No other fields are highlighted. These and other productivity features are described in 2.5.1, “Interface productivity features” on page 58.

Figure 2-20 shows the Verify interface in Datacap Desktop.

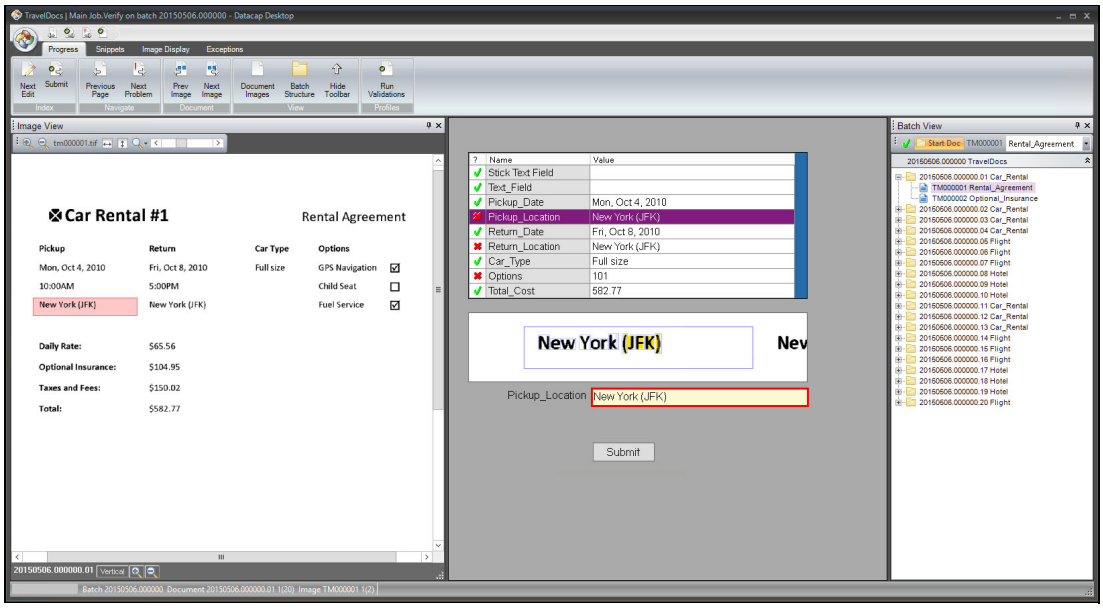


Figure 2-20 Datacap Desktop Verify Interface

Datacap Desktop customization

Datacap Desktop provides standard user interfaces, or panels, for the Scan and Verification tasks. However, the look and feel and the functions of these panels can be customized to suit your business and organizational requirements. Datacap Desktop customization is explained in detail in Chapter 12, “Customizing Datacap” on page 277.

2.5.4 Datacap FastDoc

Similar to Datacap Desktop, FastDoc is a thick client that is used to scan or import images, classify documents and fix batch structure issues, and verify captured data. It can run while connected to a Datacap server on the local area network and in *local*, or *stand-alone*, mode. When in local mode, it can send images and metadata to the Datacap server through web services after scanning and indexing is completed.

To configure FastDoc for use in stand-alone mode, the Datacap application also needs to have the Datacap capture application installed locally.

Figure 2-21 shows the Verify panel in FastDoc.

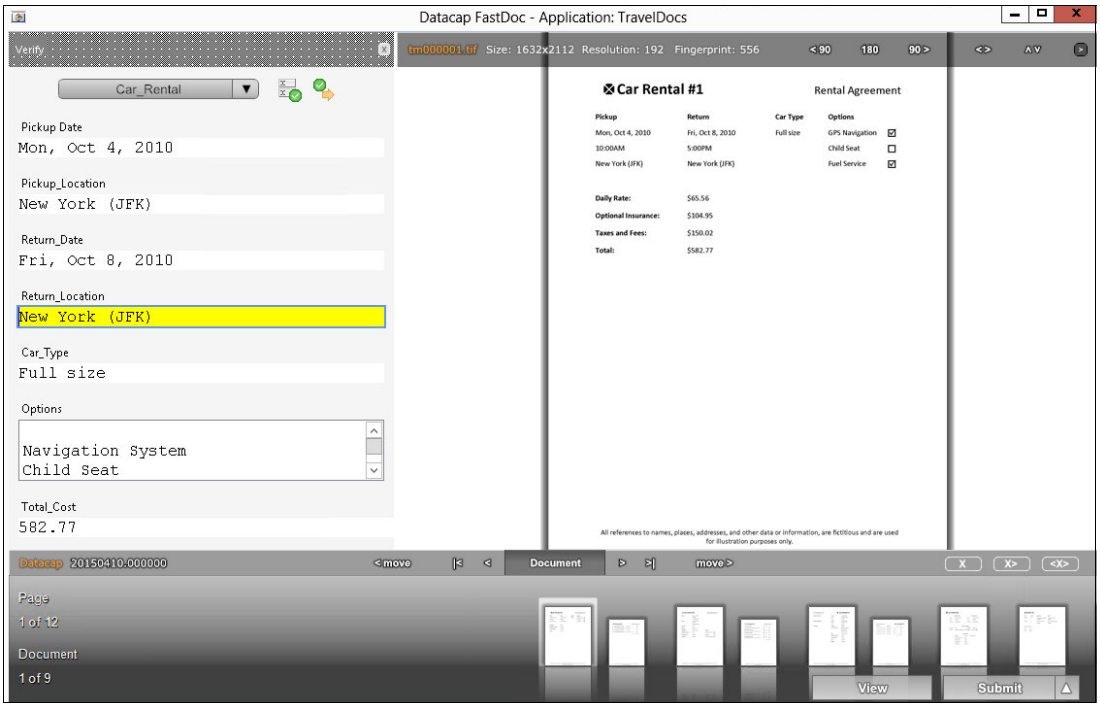


Figure 2-21 FastDoc Verify panel

2.5.5 Datacap Mobile

Datacap Mobile functions enable you to capture one or multiple documents and upload them for further processing to a Datacap server.

For example, a claim adjuster can use his iPhone or iPad to capture signed claim forms and photos of the damaged vehicle at a claimant's home to help expedite the process and close the case sooner, and to reduce the need for central scanning operations. In other instances, such as in warehouses, workers are constantly on the move checking inventories, so they need the capability to capture labels, bar codes, and so on from products on palettes.

More specifically, Datacap Mobile includes the following capabilities:

- Secured (HTTPS/SSL) and authenticated connection to Datacap applications, on-device encrypted credentials, and "sandboxed" app design to isolate app data and code execution from other apps.
- Opening of a capture in-tray, essentially a local storage space to hold and manipulate the captured images before they are uploaded to the Datacap server. Multiple capture sessions, or batches, can be saved off line and queued up for upload, which is critical when experiencing poor cell coverage.

- Capturing documents using the device's camera to point to and snap color or black-and-white photos. Two capture modes are available: Automatic and manual.
  - In automatic mode, Datacap Mobile detects the edges of documents and captures and rectifies images automatically as soon as the photo that shows in the video camera screen meets optimal quality criteria (focus, exposure, size, angle of incidence, and so on). This way, you can, for example, simply arrange the pages that you want to capture in a table and snap them by briefly panning and holding the camera over the documents.
  - In manual mode, the user triggers the camera shutter manually and captures the photo as it appears on the camera screen. This mode is intended for adding unaltered photos to the In tray, by-passing all image processing functions. This is useful when, for example, an auto insurance adjuster needs to snap photos of the damaged car.
- Importing pre-existing images from the camera, with automatic deskewing.
- Manipulating the captured images: Zoom, rotate, deskew, crop, adjust brightness and contrast, add, delete, and reorder images.
- On-device optical character recognition to automatically extract textl content from images and populate document data fields that can be presented to the user for verification and repair.
- On-device 1D and 2D bar code recognition to automatically populate data fields or assist with automatic classification.
- Classifying and assembling images into documents that can be indexed with business data entered manually or extracted using OCR and bar code recognition.
- One-click application setup from a link sent through email. Connectivity to the server-side Datacap application is configured once by entering the server's URL, Datacap application, credentials, and mobile profile, all preset on the Datacap server side. After completing and testing the server configuration, it is sent as a link by email to other users.

A comprehensive overview of Datacap's mobile capabilities is provided in Chapter 11, "Datacap Mobile user experience" on page 257.

2.6 Application configuration

At a high level, the principles for setting up a Datacap application are easily understood if you visualize a document and the types of data that you are trying to extract from it. A document is made up of pages that are typically identifiable by certain characteristics. Such characteristics include a specific structure, the layout of each page, and the location in the page of the specific pieces of information that you are seeking to extract.

Configuring an application consists of providing Datacap with a combination of visual clues and processing rules from its catalog of rulesets and actions. They guide how to automatically recognize and separate the documents; find, capture, and process the data in them; and transfer both images and data to the back-end systems.

Configuring an application involves these steps:

1. Use FastDoc to model and build most of your application. Use *compiled rulesets* and *application templates* to quickly create your application.
2. Test the runtime batch processing with FastDoc, tweak workflow and jobs with Navigator Admin, and configure interfaces (Mobile, Navigator, FastDoc, Desktop) for user-attended tasks.

3. Use Application Manager for background task profiles that are run by Rulerunner and other application-wide options, and configure Rulerunner.
4. Test the application end-to-end within the target user interfaces.
5. Deploy your application by using the Application Copy Tool.
6. Set up Datacap Maintenance Manager.
7. If necessary, extend rulesets and use Datacap Studio to customize them with additional actions.

Datacap applications can be further extended by using Visual Studio to customize Desktop panels, developing custom actions and new compiled rulesets, and custom reports.

Custom mobile app for iOS and Android can be developed using the Datacap Mobile SDK.

Each of these configuration tasks is described in detail throughout the remainder of this book.

3

Advanced imaging architecture overview

This chapter provides an overview of the IBM Datacap architecture. Several architecture options are described, with typical use cases outlined for each option.

This chapter covers the following topics:

- Architecture overview
- Components of the Datacap system
- Deployment patterns

3.1 Architecture overview

Datacap components fall into three categories: base, supporting, and optional.

Base components are integral to Datacap. Supporting components are external components that Datacap must interface with, such as databases and file systems. Some supporting components are required for Datacap to operate. Optional components are not required to operate Datacap but are available. Optional components can provide services such as external content repositories or authentication services.

Figure 3-1 illustrates Datacap components at a high level.

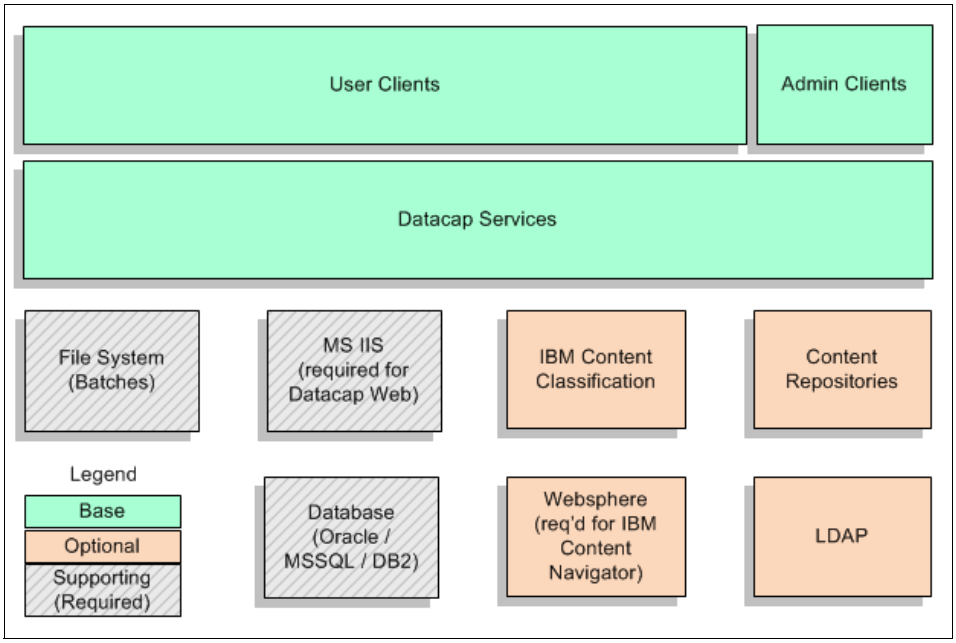


Figure 3-1 High-level architecture

Figure 3-2 shows a more detailed view of the Datacap system architecture.

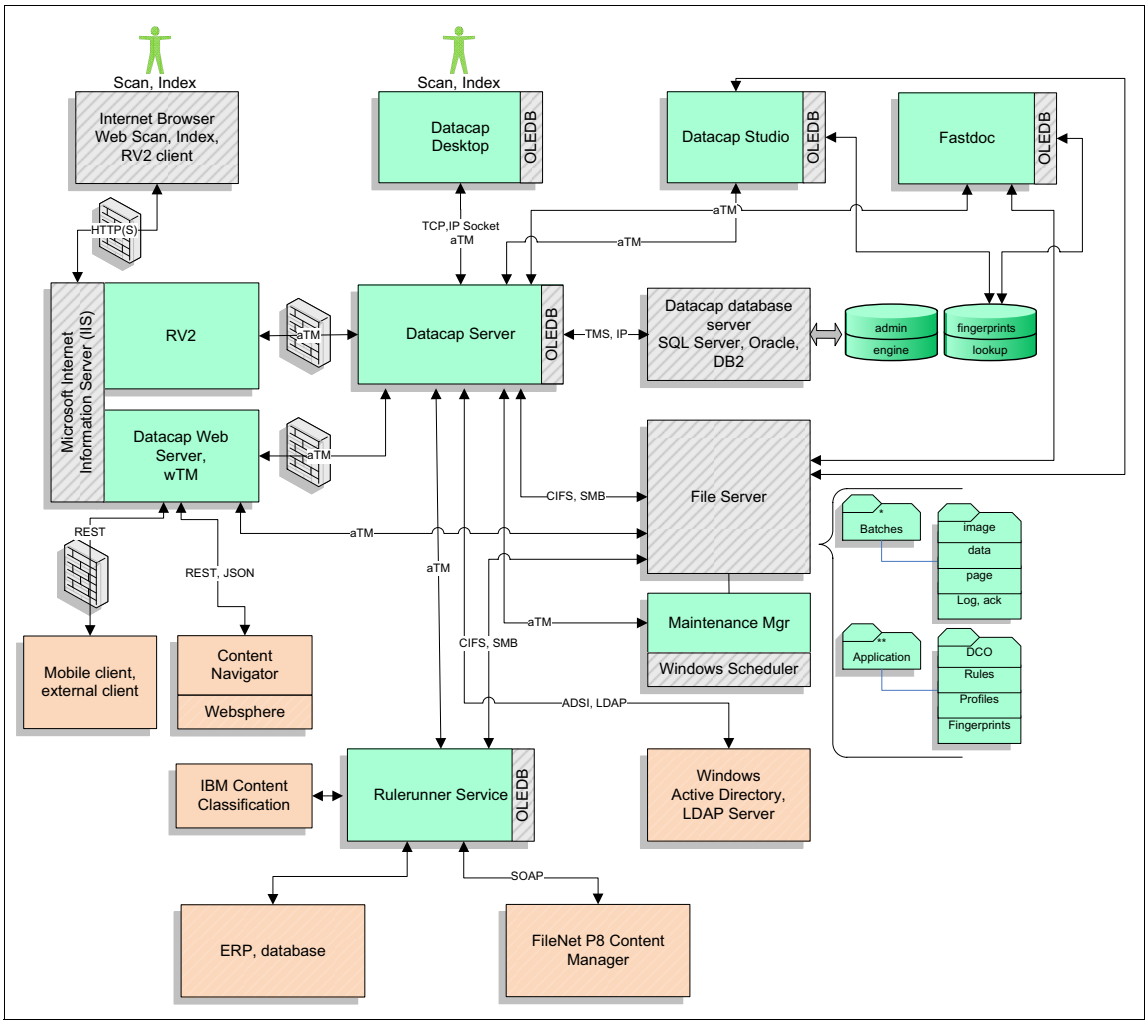


Figure 3-2 Logical architecture diagram

3.2 Components of the Datacap system

Components of the Datacap system include server-side components, administrative clients, and user clients. The sections that follow describe these components.

3.2.1 User clients

Datacap provides client components to address several capture scenarios and use cases:

- Datacap Desktop
- FastDoc
- Datacap Navigator
- Datacap Mobile
- Datacap Web Services

These clients are described in 2.5, “Datacap user interfaces” on page 58 and in Chapter 10, “Datacap user experience in IBM Content Navigator” on page 233.

3.2.2 Administration clients

Administrative clients are used to configure Datacap solutions, control access to applications, and perform maintenance on the system. This section provides an overview of these clients.

Datacap Studio

Datacap Studio is used to define and configure Datacap applications by defining and assembling the document hierarchies, recognition zones and fields, rules, and actions. It requires access to the file server and the Datacap databases.

Datacap Maintenance Manager

Datacap Maintenance Manager (DMM) automates recurring system health and housecleaning tasks, such as batch monitoring, status notification, and automatic deletion of completed batches. Tasks are scheduled by using the Microsoft Windows Scheduler.

DMM can execute any rulesets and actions defined for it in Datacap Studio by associating its ruleset and task profile with the applications that you want to monitor. Typically, you use DMM to perform selections in the Engine database of the application and execute actions on the selected batches.

For example, you might run the following actions:

- Monitor batches, notify statuses, and automatically delete completed batches.
- Identify batches that meet certain criteria, such as batches that stopped.
- Change the status of batches and their order in the queue.
- Delete batches or move them to another location.
- Capture data snapshots to a database to be reported by using Datacap Report Viewer.
- Send email notifications, such as of error conditions or a batch stopping.

You can run DMM in three ways:

- Manually by using the DMM Manager
- Automatically by using the Windows Task Scheduler, either at scheduled times or when triggered by a system event
- Automatically as a task of the workflow of an application

FastDoc

The FastDoc tool is a powerful new way to build applications. It uses compiled rulesets that help the operator configure an application quickly. Compiled rulesets are rulesets that offer a preset list of actions that can be configured easily through a single interface.

Figure 3-3 shows the FastDoc workflow configuration interface.

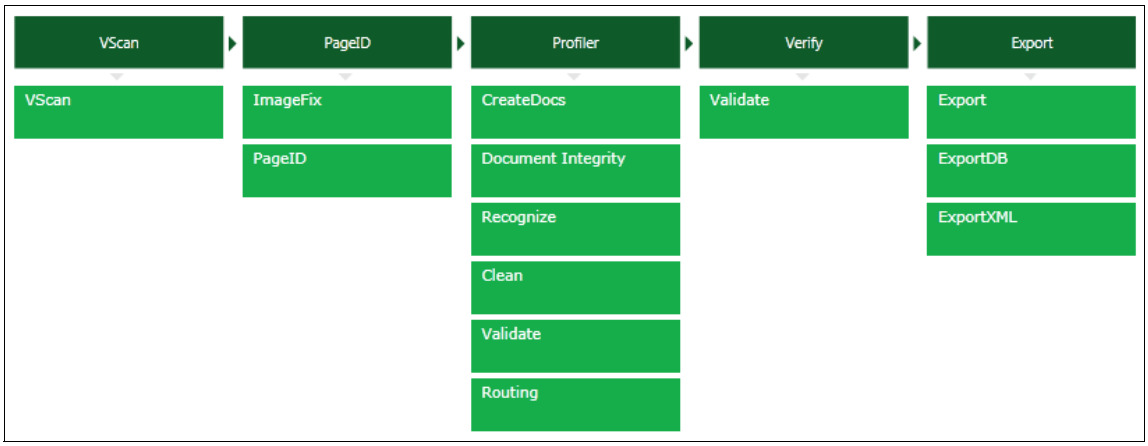


Figure 3-3 FastDoc workflow configuration interface

Figure 3-4 shows a sample compiled ruleset for the identification of page types. Using this interface greatly simplifies the configuration of classification rules.

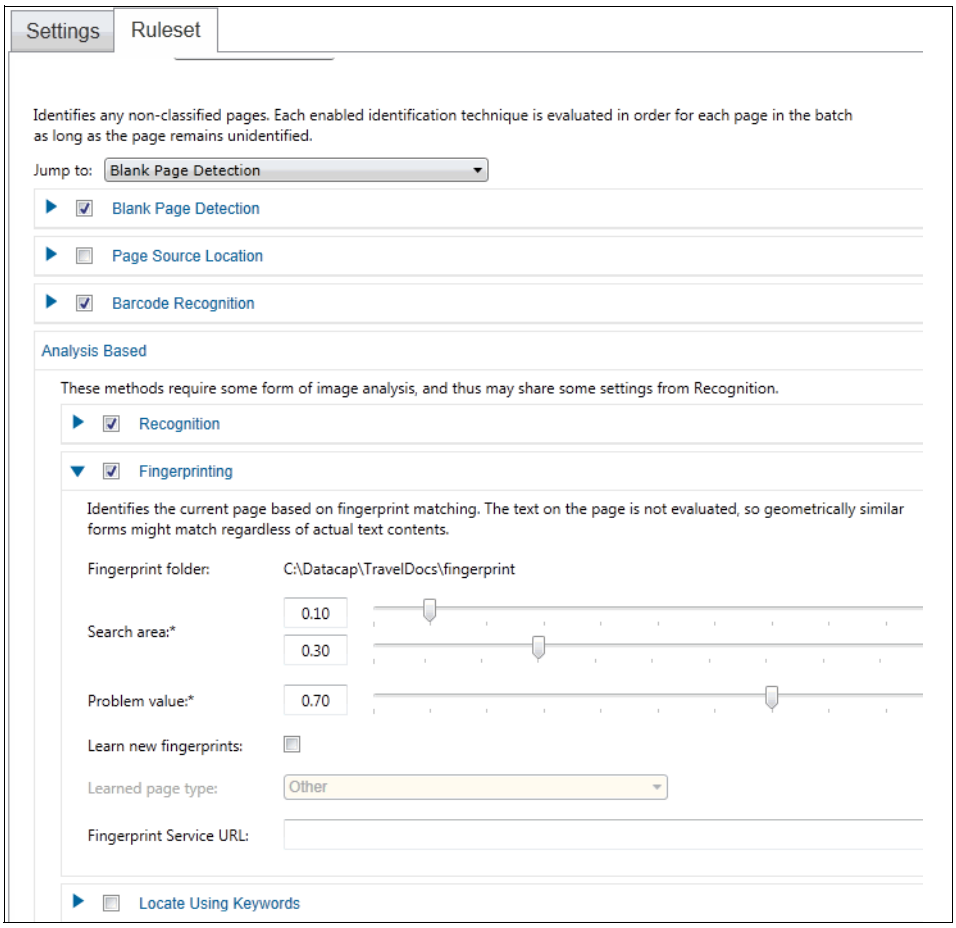


Figure 3-4 Sample compiled ruleset

Datacap Application Manager

Datacap Application Manager manages environment-specific information about the applications of a Datacap system in a central registry. This way, the Datacap system components that are running on different machines can be made aware of each other and get access to the information that they need to operate. The information includes pointers to the Datacap server, databases, fingerprint library, and file server.

Datacap Application Manager maintains a separate registry for each application, cross-referenced by the central registry. To deploy an application on multiple machines, or from a development to a production system, you move the resources of the application (configuration files, working directories, databases, and so on) to their target machines. Then, you update the registry of the application with the new locations and the cross-references between the registries of the application in the central registry.

Datacap Application Manager, shown in Figure 3-5, is typically installed in restricted (read-only) mode on all workstations, except the machine that is running Datacap Studio, to prevent users from modifying their deployment settings. In restricted mode, only the reference to the central registry can be modified.

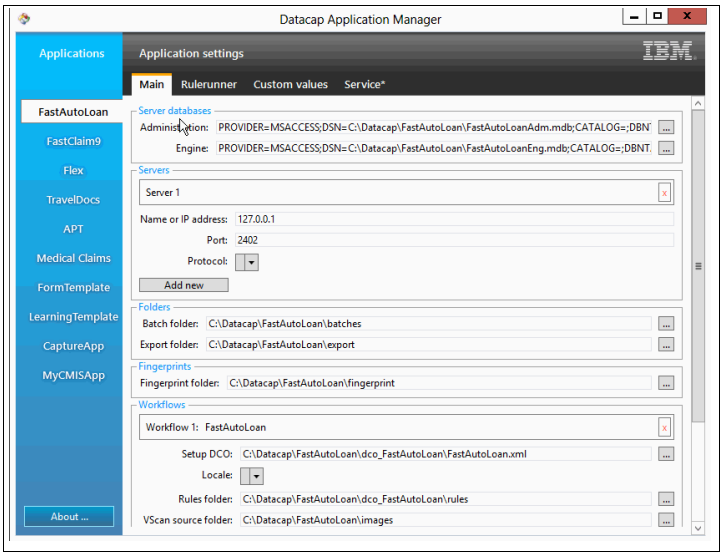


Figure 3-5 Datacap Application Manager

In Datacap Application Manager, for each application, you can define which tasks you want Rulerunner to run automatically in the background. You can also store application-specific custom values, such as a connection string to a lookup database or even credentials. These values can be retrieved and passed at run time to Rulerunner actions to avoid hardcoding this information in the rules and make them more portable and secure.

Datacap Application Copy Tool

The Datacap Application Copy Tool (DACT) provides an easy way to copy or migrate an application from one environment to another. For example, this tool can be used to simplify moving an application from a development environment to a test environment. The DACT can also be used to migrate the databases from one database provider to another, such as from Microsoft Access to Microsoft SQL.

3.2.3 Datacap services

Datacap services perform specific background tasks to support the various deployed solutions. In this section, we outline these services.

Datacap server services

The Datacap server is the heart of the Datacap system. It manages and serves batches to workstations and users. It also orchestrates the tasks according to the workflow of the Datacap application. It provides user authentication and access control, assigns batch IDs, controls batch queuing, and controls access to the Datacap databases.

All communications between the Datacap server and its clients or the other core Datacap server components use the Datacap socket protocol. For communicating with the databases, it uses Microsoft Object Linking and Embedding for Database (OLE DB). It uses the Common Internet File System (CIFS) interface to mount the file share that is required to access batches. Datacap server also uses Active Directory Service Interfaces (ADSI) or LDAP to communicate with the Directory Service for user authentication.

Datacap Rulerunner services

The Rulerunner service runs all tasks that do not require operator intervention, such as image cleaning, conversion, recognition, classification, and export to content repositories, such as FileNet Content Manager, IBM Content Manager, and other IBM Content Management Interoperability Services (CMIS) compliant repositories. Datacap Rulerunner interfaces with FileNet Content Engine through the Datacap Web Services application programming interface (API).

Fingerprint services

The Fingerprint Service is a web server that stores the field locations for all active fingerprints. Without it, a background computer must load the zones from the network every time a batch is processed, which can add significant time to the background processing.

Datacap Web Services

Datacap Web Services (previously called Taskmaster Web Services, or wTM) is a REST-based web service used for communicating with Datacap. It gives a remote device or system the ability to create and trace a batch through the IBM Datacap capture process.

The Datacap Web Services API supports the following activities:

- Creating a new batch
- Uploading pages to a batch
- Setting the page file name
- Updating page files
- Releasing a batch to the next task
- Retrieving any file in the batch folder
- Retrieving batch information such as batch ID and batch status

Report Viewer

Datacap Report Viewer (previously called RV2) is a web application that is used to display Datacap reports on system activities, such as batch status, station activity, or problem batches.

Reports can be filtered to display areas of interest. Custom reports can be configured by using Microsoft Windows Forms. It is also possible to build custom reports by using commercial reporting systems that are capable of querying Datacap report database tables.

3.2.4 IBM Content Classification

IBM Content Classification categorizes and organizes content by combining multiple methods of context-sensitive analysis.

Datacap provides recognition results to Content Classification, which examines and analyzes the full text of these documents and returns the suggested page type back to Datacap. The system trains itself with actual examples from the organization, and it can learn from user feedback and incorporate that feedback into the system's understanding in real time.

3.2.5 File server

A file server hosts image files, extracted data and control files, and files that are required for running various applications, such as the fingerprint files and document hierarchy definition files. The file server must be shared across all the components that need to process the batches. Fast access to the file server is essential to ensuring high performance of the Datacap system.

3.2.6 Microsoft IIS

Datacap Web, Datacap Web Services, and Report Viewer are installed on Microsoft IIS.

3.2.7 Database

For its operation, a Datacap application relies on the following relational databases that are hosted in a production system in IBM DB2, Microsoft SQL Server, and Oracle:

- The Administrator database stores information about users, groups, workstation, auditing, functional security, and application configuration. It also stores workflow configurations.
- The Engine database stores information about batches, statistics, and queue states.
- The Fingerprints database manages the pointers to the fingerprints that are used in a specific application.

Each application has its own set of self-contained databases. It is possible to consolidate multiple administrator and engine databases. Also, in many cases, Datacap applications need access to other databases to perform lookups or export data to line of business (LOB) systems and databases.

In Datacap sample and add-on applications, Microsoft Access database is also used for portability reasons but must not be used in production.

3.2.8 Content repository

Content repositories are commonly used to store the scanned images and the associated metadata. Datacap can export to IBM Content Manager V8 (CM8), FileNet P8, IBM Content Manager On Demand (CMOD) and non-IBM repositories, such as Microsoft Sharepoint and Content Management Interoperability Services compliant repositories.

3.2.9 LDAP

An LDAP or Active Directory service is also often part of the configuration for Datacap users to authenticate as an alternative to native Datacap authentication.

3.2.10 IBM WebSphere Application Server

IBM WebSphere® Application Server is used to host the optional IBM Content Navigator web client. IBM Content Navigator communicates with Datacap through Datacap Web Services. Although WebSphere can be installed on several different operating systems, if you want to collocate Datacap Web Services on the IBM Content Navigator server, make sure that you install it on a supported Microsoft Server operating system. You can find supported Microsoft Windows operating systems by searching the “Detailed system requirements for a specific product” web page:

http://ibm.co/1gf1BOF

3.2.11 Connection to a business application or database

Typically, connection to a business application or database is through Open Database Connectivity (ODBC). For example, a customer, vendor, or purchase information can be queried against a database and used for image process verification purposes. Information that is extracted from an image can be exported to business applications and databases.

3.3 Deployment patterns

In this section, we describe different deployment patterns. It is possible to deploy hybrid configurations according to organizational requirements.

We cover the following deployment patterns:

- Centralized deployment
- Regional deployment
- Web deployment
- Local deployment
- High availability, load-balanced deployment

Each of the deployment patterns described is only a sample representation of what an environment could look like. It shows the granularity in which the services can be broken out. In many scenarios, organizations opt to collocate some of the services. Additionally, Datacap customers commonly use virtualization technologies to reduce the physical footprint of the deployment while maintaining the separation of services.

3.3.1 Centralized deployment

Centralized deployments are used when operations must be concentrated in one place, such as in a traditional mailroom scenario. This approach is best suited when incoming image volumes are high and when economies of scale can be derived from pooling resources and specializing operators to specific tasks, similar to an assembly line.

In this scenario, Datacap servers and users are located in a single location. Both Scan and Verify functions are available through either thick or web client. For web clients, organizations can deploy Datacap Web and IBM Content Navigator. For Datacap Web, Report Viewer, or Datacap Web Services running as a IIS application, organizations must install Microsoft IIS.

With Content Navigator, you must install WebSphere Application Server. IBM Content Navigator connects to Datacap through Datacap Web Services. You can collocate Datacap Web Services on the IBM Content Navigator server if it is installed on a supported Windows server. If IBM Content Navigator is installed on a different platform, you have the option of setting up a dedicated Datacap Web Services server or installing Datacap Web Services on a Datacap Web server. However, you need to make sure the sizing has been reviewed carefully to ensure the IIS server does not get overwhelmed.

The sample diagram, provided in Figure 3-3 on page 71, will need to be tailored and sized to your organization's specific requirements. For example, you might need to split servers more granularly or combine some servers.

Note: With version 9 of Datacap, it is possible to run Datacap Web Services as a Windows service eliminating the IIS dependency when using custom web clients or IBM Content Navigator.

Figure 3-6 shows an example of a centralized environment deployment.

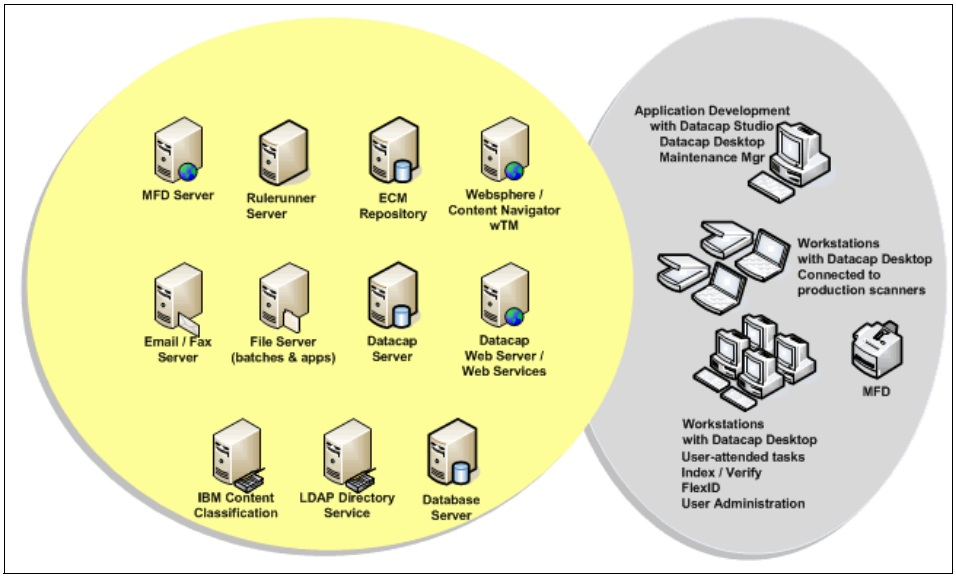


Figure 3-6 Centralized deployment

Multifunction device and printer integration

Support for multifunction devices or printers (MFDs or MFPs) is accomplished through NSi AutoStore or Imagine Solutions' Encapture applications. These solutions might require additional hardware. Your IBM sales team can help you select the best solution based on your specific needs.

NSi AutoStore

NSi AutoStore uses Datacap Web Services to pass document images and metadata to Datacap or directly to the content repository. NSi AutoStore communicates with Datacap through Datacap Web Services.

Figure 3-7 shows a single-server deployment of NSi AutoStore.

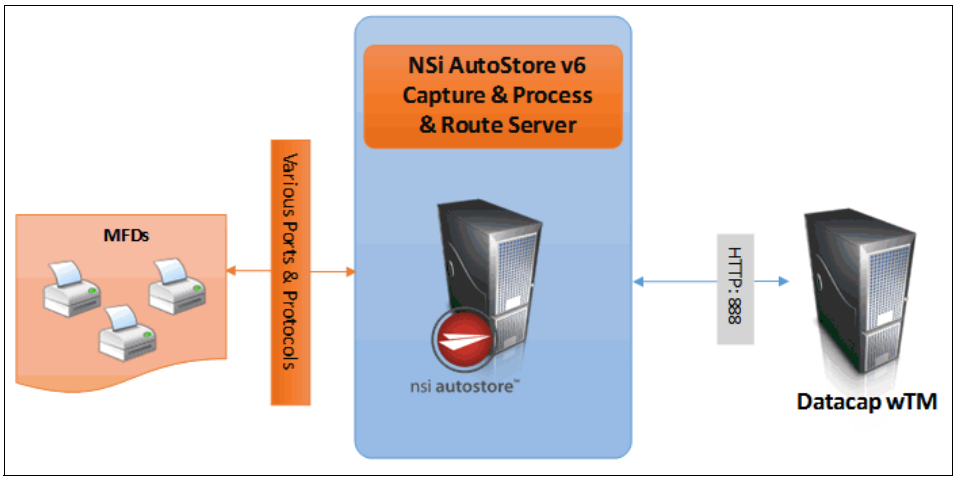


Figure 3-7 NSi AutoStore - single-server deployment

The NSi AutoStore server installation carries required modules for system deployment: The AutoStore service module for capture, process and route, the process designer, and status monitor.

All processes in AutoStore consist of the following layout:

Capture	Obtains documents, files, and control devices
Process	Handles recognition, image management, and conversion
Route	Stores the documents in the specified repository

The system is configured in a graphical workflow designer, which eliminates the requirement for scripting, and enables new processes to be quickly created.

Imagine solutions: Encapture

Encapture stores the image files in a network folder, along with indexing metadata, in an XML file. Datacap applications can then retrieve data from these documents by adding the Encapture scan actions to the appropriate applications.

Figure 3-8 shows the Encapture Scan action that is used.

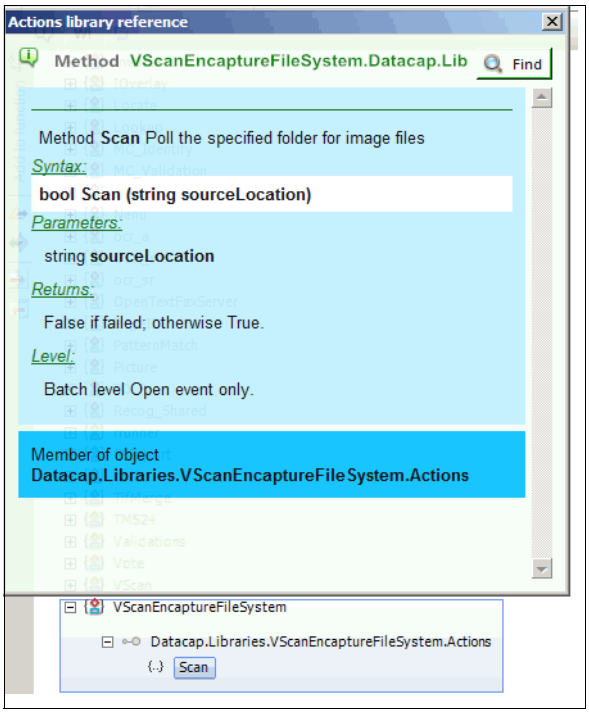


Figure 3-8 Encapture SCAN action in Datacap Studio

Encapture requires the following services, which are often collocated on a single server:

- Discovery service

  The Discovery service is primarily responsible for importing batches that originate from an external system, such as fax servers, multifunction devices, and other capture applications into Encapture. To support these, Discovery is extensible through a connector architecture. Discovery can be configured to invoke a specialized connector for each configured external batch source, which formats the metadata of the external source to Encapture native format.

- Delivery service

  The Delivery service packages all batch data and images and presents them to the target system. The target system is selected based on the batch content type of the batch. This is the last step in the Encapture system and is actually the export from the Encapture system. Delivery is extensible through use of connectors that perform the actual mapping, packaging, and delivery of data and images. Typically, an Encapture system exports to a repository or image-processing application. Multiple connectors can be used to connect to multiple back-end systems, although a single batch will be processed by only one connector.

- Process service

  The Process service classifies document, extracts data, and cleans up images on Encapture batches. It uses the Encapture workflow system to lock batches for processing and delivers the batch to the appropriate Datacap Rulerunner service and application based on the batch content type.

- Cleanup service

  The Cleanup service deletes image files and database records from the central batch store, based on the batch store retention schedules. It also deletes any expired session records.

Figure 3-9 shows a typical deployment of Encapture on a single server.

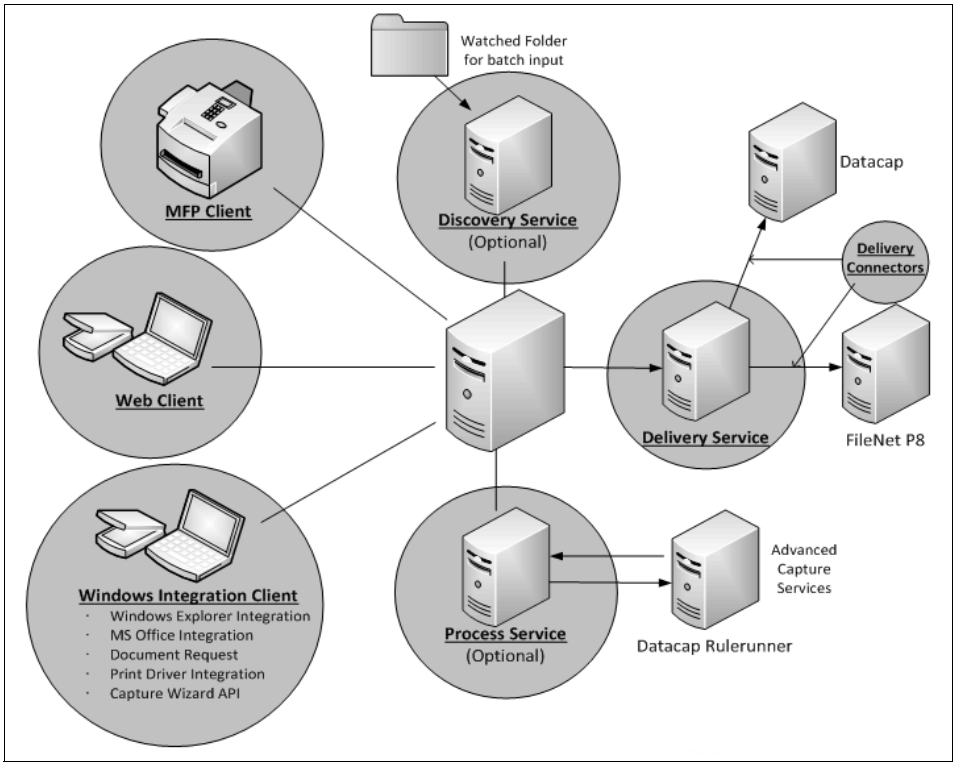


Figure 3-9 Typical Encapture deployment

3.3.2 Distributed deployment

A distributed deployment configuration is ideal for organizations with geographically dispersed user populations and resources where key system resources can be located closest to users. A distributed deployment can be thought of as a variant of the central deployment model. In a distributed deployment, a large population of users and sizable capture operations justify installing system resources in regional offices.

For example, in a regional office, you might want to install an instance of Datacap and a departmental scanner.

Distributed deployments are typical in organizations with these requirements:

- Scan documents from multiple locations.
- Scan centrally but need to have remote users verify the documents.
- Use outside vendors to scan or verify images.
- Use mobile capture and indexing capabilities.
- Have remote multifunction devices and printers that must participate in captures.

Consider the following factors when planning for a distributed deployment:

- Bandwidth

  Bandwidth is always at the top of the list of considerations. Insufficient bandwidth can make it difficult to perform indexing and verification tasks, because images must be uploaded to the verification station in real time.

- Hours of service

  Some tasks might not need to be done in real time. For example, it is possible to scan during the day and upload documents to the Datacap servers during off hours to reduce the load in the network. When possible, a strategy of using schedules for certain tasks is advised.

Regional deployment of Datacap

This deployment option requires that a Datacap instance be installed in the regions. This is ideal for situations where the scan and index operations must be performed in the regions and the volumes make it difficult to process by using web clients. Again, a choice can be made whether to deploy IBM Content Navigator.

Batches can be processed entirely in the regions with the export to the central site content repositories run after hours when bandwidth is maximized. The number of servers required in the regions varies, depending on the volumes to be processed.

Figure 3-10 shows an example of a multi-region deployment configuration. In this example, there are three deployments of Datacap, each configured differently, all exporting their documents to a central ECM repository. In one region, we show Datacap installed on a single server. Remember to size your environment adequately before making a decision to collocate installation components.

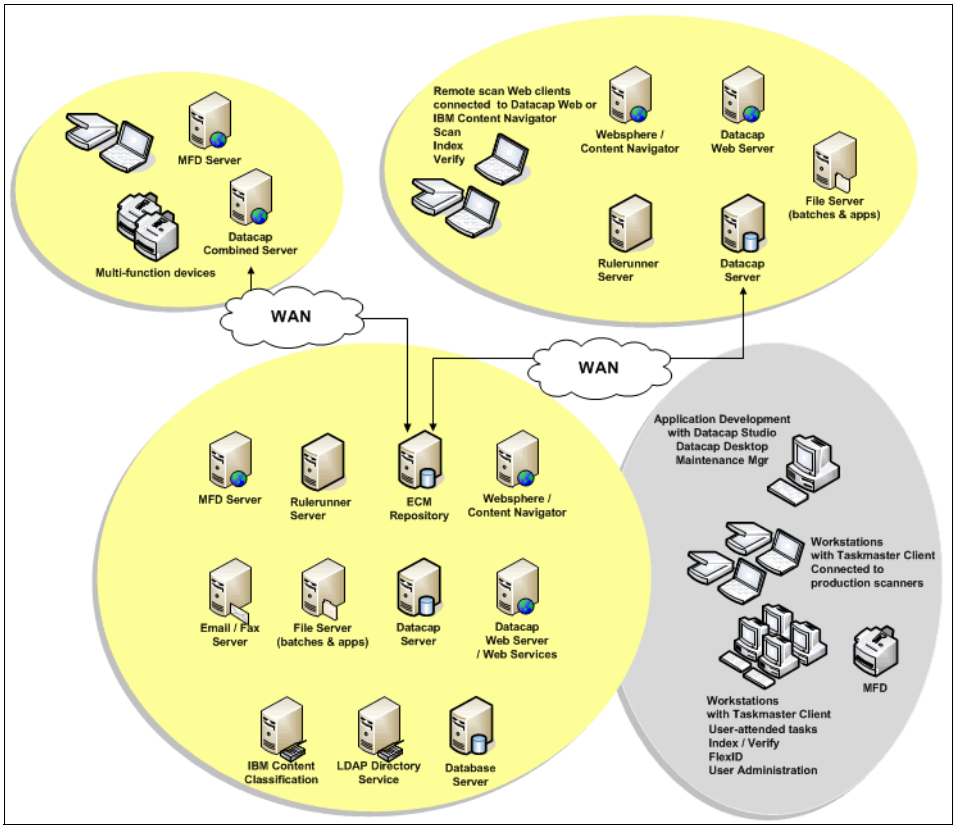


Figure 3-10 Regional deployment

3.3.3 Datacap Web deployment

This section describes the Datacap Web deployment patterns.

Central server, remote web clients

In this scenario, the Datacap servers are all located centrally. Clients use one of the web client options for Scan, Fix, and Verify operations.

Figure 3-11 shows a typical web deployment with multiple remote sites. In one remote site, users scan using MFDs that will upload the scanned images to the MFD server. The second site uses Datacap Web or IBM Content Navigator to perform operations such as scan, index, and verify. This architecture provides an environment where scanning and verification tasks can be performed remotely while using a centralized server farm.

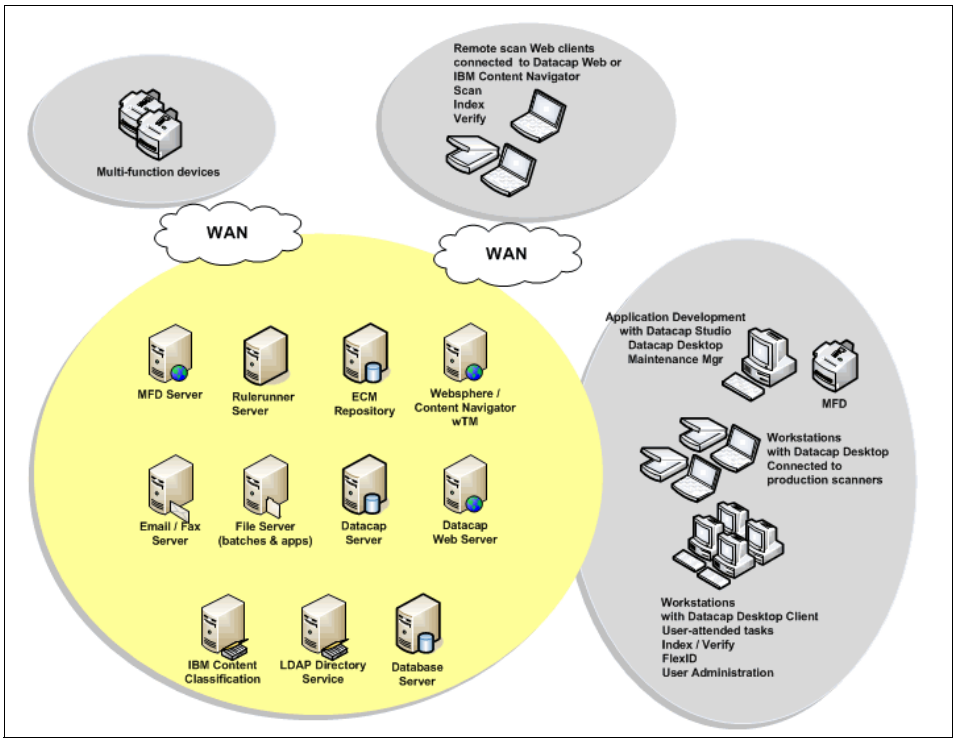


Figure 3-11 Web architecture diagram

Flexibility can be added to the previous two deployment scenarios with Datacap Web. Datacap Web provides close to the same functions that are available in its thick client, including scanning, importing, indexing, and verifying documents, in addition to administering the Datacap system. Essentially, all user-attended functions of the typical Datacap process can be performed through a browser.

For example, by adding Datacap Web to the deployment scenarios described earlier, you can perform the following tasks:

- Supplement the indexing and verification operations for documents that have been scanned at the central location by using resources in remote locations. This task is ideal in situations where remote users are most familiar with the content being processed or where additional assistance is required to handle peak scan volumes.

- Distribute “document-at-a-time” scanning at the source while having the indexing and verification done centrally. For example, in this scenario, shipping personnel scan documents locally but the indexing and verification are executed centrally, where more customer information might be available from the LOB systems.
- Offload all scanning, indexing, and verifying operations to the local offices. These offices have all the information necessary for these operations and are most likely to use the documents after they are committed to the content repository. This task is possible if the volumes for each individual are manageable. In this scenario, you do not need many resources in the central location beyond simply monitoring and maintaining the systems.

Although using Datacap Desktop might possible for remote users, it is usually preferable to use one of the Datacap web clients. Desktop clients require a high level of connectivity, which is often difficult in distributed environments. Web clients, instead, perform extremely well in these types of environments.

Datacap web clients fall into three categories; Datacap Web, Content Navigator, mobile and Datacap Web Services custom clients.

Browser client

Here, we introduce two of the Datacap browser clients.

Verifine web client

The Verifine client is a configurable client. With Verifine, it is possible to modify the layout of the panels to suit your organization’s preferences.

Figure 3-12 shows the configuration options of the Verifine client.

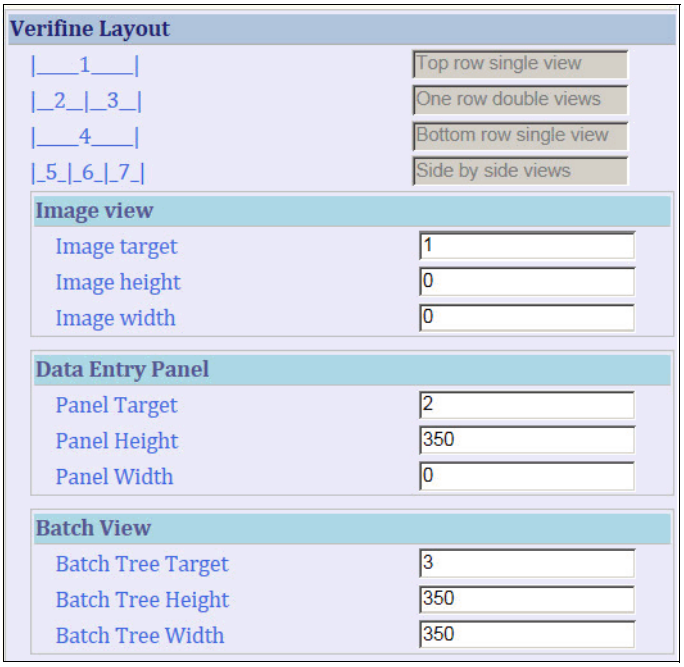


Figure 3-12 Verifine client configuration

aVerify and aScan web clients

The aVerify and aScan clients are ideal for situations where bandwidth is at a premium. Both clients (one offering scanning services, the other verification and indexing services), use Ajax technology to limit the amount of information that needs to be transmitted between documents making it faster.

Content Navigator

IBM Content Navigator provides both user functions, such as scanning and verification interfaces, and administration functions. IBM Content Navigator provides drag interface designer capabilities. See Chapter 10, “Datacap user experience in IBM Content Navigator” on page 233 for a detailed overview of Datacap in IBM Content Navigator.

3.3.4 Local processing

Alternatively, in some specific circumstances, it could be possible to install Datacap services locally on a workstation if it meets the minimum supported configuration. Although this type of deployment is rare, it could be ideal for situations where volumes are low enough that they do not need server configurations.

For example, a small regional office that scans a small number of documents might want to perform the validations immediately. Having Datacap installed on a single workstation helps simplify deployment and reduces hardware costs. In this scenario, you could have the same deployment on multiple workstations, which provides a level of redundancy if one workstation becomes unavailable.

One option for this type of deployment is to use the Datacap FastDoc application. FastDoc is easy to configure and deploy and can be run either in local or Rulerunner mode.

Consider these caveats with using this type of deployment:

- Each workstation must have a copy of the application.
- Changes to the application need to be replicated to all workstations.
- Fingerprints are not shared among the workstations, although they could be migrated from one system to another.
- All software upgrades or patches must be installed on each workstation.

3.3.5 High availability and load balancing

Although it is possible to run all Datacap components on a single server, it is rarely done for several reasons, which include the need for redundancy and scalability. In this section, we describe high availability and load balancing options in Datacap version 9. For the purposes of this description, we refer to high availability and load balancing as simply “load balancing.”

Load balancing is a method for scaling a system horizontally by distributing the work across many compute nodes in a “farm.” It also provides high availability by redirecting clients to a working node in case of failure. A load balancer presents a single address for communication with multiple servers, for one or more Datacap applications. Configure the load balancer to send requests that are directed to each pooled or balanced address to one of the servers in the farm. You can select round-robin scheduling or another method.

Clients access the Datacap server by using a TCP/IP socket-based protocol. You configure the server's name or IP address and port in Datacap Application Manager. The server normally listens on port 2402, but you can change the port in Datacap Server Manager. In that case, you must also configure the port number in Datacap Application Manager.

The socket keeps the TCP/IP session active until the client disconnects. As a result, the load balancer connection for Datacap server must not be persistent. If a load-balanced server fails, further client requests will be directed to a different server. In that case, the older session will be invalidated and the user or client will have to log in again. Any outstanding server requests will terminate unsuccessfully, and any batches that were in process using that server will typically be left in running status. Users who logged in to this server will receive an error message and must log in again. Datacap Maintenance Manager can be used to reset batches left in error state so they can be processed by another server.

Datacap Web Servers can also be farmed. Designate one or more IP addresses or ports on your network for your Datacap website home pages. Client browsers connect to this load balancer port using HTTP or HTTPS protocol. Configure your load balancer to redirect those requests to individual web servers using round-robin scheduling or another method. Ports 80 and 443 are standard. You can configure an alternate port in Microsoft IIS Manager.

Datacap Web uses session cookies, so you must configure the load balancer to persist sessions based on the client's IP address. Set the load balancer's session time-out to match the IIS session time-out. If a Datacap Web Server fails, users who connect to the failed server receive an error message and must log in again.

Datacap Web Services can run on IIS or as a Windows service and can be farmed. Clients connect to the address and port for the load balancer and are redirected to a specific server. Sessions must be persistent, based on the client IP address, and the session time-out must match that of the web service. Failure of a web services server will generate an error for requests in progress. The requested operations might or might not have completed.

Datacap Report Viewer servers can be farmed, Clients connect to an address and port for the load balancer and are redirected to a specific server. Sessions must be persisted, based on IP address, and the session time-out must match that of the IIS server. Failure of a Report Viewer server will end any existing sessions.

Datacap Fingerprint Service can be farmed if the fingerprints are static during normal system operation. Updates and deletions of fingerprints are not synchronized automatically between servers. Fingerprint servers must either be restarted or their contents programmatically reset to keep them synchronized if changes are made to the set of fingerprints.

Datacap Rulerunner Servers independently poll Datacap servers for pending work, they do not require or benefit from load balancing. To achieve redundancy of servers, threads should be duplicated on at least one additional server. For example, if Rulerunner server A has a thread running for Application 1, Profiler and Export, setting up an Application 1, Profiler and Export, thread on server B provides redundancy.

Table 3-1 summarizes load balancing options for Datacap servers.

Table 3-1 Load balancing options

Server	Load-balanced	Protocol
Datacap server	Yes	TCP/IP socket
Datacap Web Services	Yes, persistent connections	HTTP/HTTPS
Datacap Web (IIS)	Yes, persistent connections	HTTP/HTTPS
Report Viewer (IIS)	Yes, persistent connections	HTTP/HTTPS
Fingerprint server (IIS)	Yes, persistent connections	HTTP/HTTPS
Rulerunner	Yes	
Content Navigator (with WebSphere Application Server)	Yes	HTTP/HTTPS

Grant network access to the back-end server addresses if possible. This makes initial setup and subsequent problem solving much easier. Test your system without load balancing at first. Add load balancing to one component at a time, reconfiguring as needed, and testing each balanced address, including failover to each back-end server, before moving on to the next component. If policy requires that you disable connections to the back-end servers, be prepared to re-enable for troubleshooting, if required.

Figure 3-13 illustrates a sample load balanced architecture.

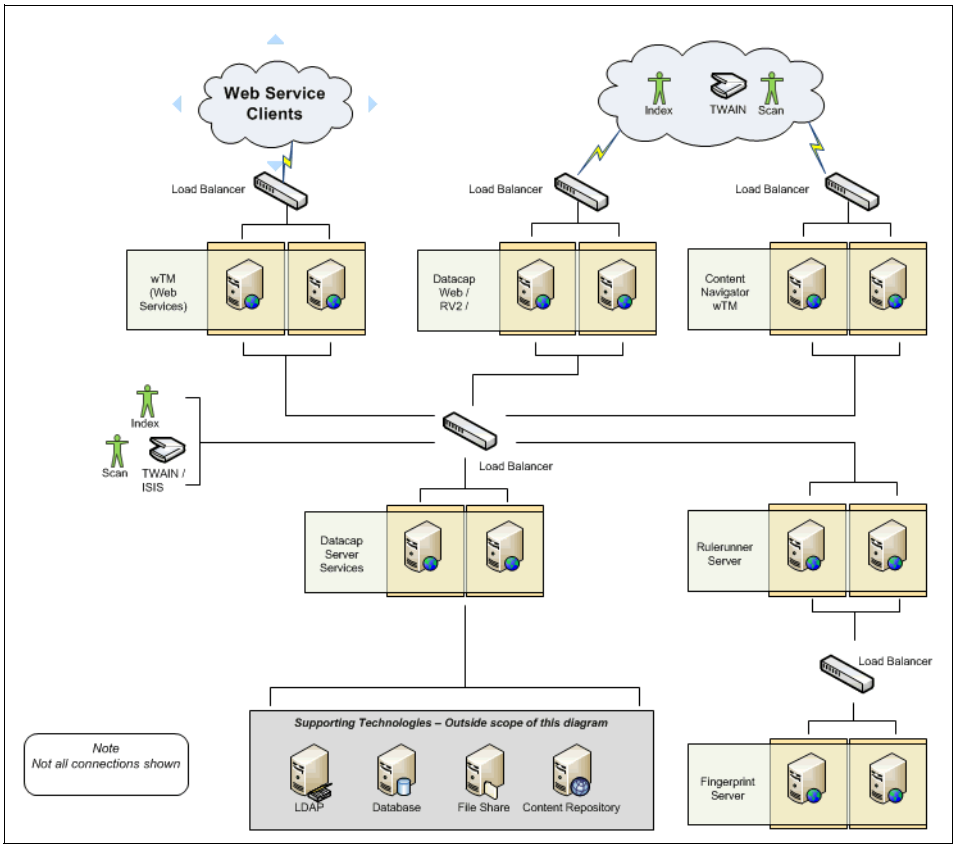


Figure 3-13 Sample load balanced architecture

For more information, see the IBM Technote titled “Load Balancing and Farming Datacap Servers”:

http://www.ibm.com/support/docview.wss?uid=swg21668810

4

Planning considerations

This chapter describes considerations for implementing an enterprise IBM Datacap solution. The primary focus is on the planning, design, and deployment, specifically related to the tools for handling discovery, requirements gathering, and functional design.

This chapter includes the following sections:

- Set goals for the enterprise imaging solution
- Define requirements for the capture system
- Gather requirements

4.1 Set goals for the enterprise imaging solution

Exploring potential production imaging projects begins with identifying business goals. At a high level, business goals can include the following examples:

- Reduce costs.
- Shorten the business process cycle time.
- Improve service.

If the current process uses paper documents, you can eliminate or improve many of the following manual tasks:

- Receiving documents, logging, counting, batching, and date-stamping
- Sorting documents for filing and distribution
- Preparing file folders
- Filing documents
- Distributing documents for processing
- Photocopying for distribution
- Manual typing of data
- Retrieving files from file cabinets
- Searching through files to find documents
- Matching documents against exceptions reports
- Re-filing documents and files
- Pending and suspense file management
- Keeping calendars or diaries to track follow-up documents
- Searching for misplaced and lost files
- Reconstructing of lost files
- Purging files and removing selected documents for disposition
- Transporting documents to and from storage rooms or off-site storage
- Filing internal forms or copies of correspondence

In addition to eliminating and improving manual tasks, eliminating paper offers other potential savings, such as the following examples:

- Storage-space savings from eliminating file storage areas
- Office space savings (including lighting, heating, furniture, and so on)
- Archive filing costs onsite and off-site
- Reduced workstation equipment and support costs
- Reduced filing equipment costs
- Reduced number of microfilms, cameras, processors, viewers, and consumables
- Reduced number photocopiers
- Reduced equipment maintenance of all types listed previously

4.2 Define requirements for the capture system

You must consider the requirements that are unique to the area of capture. This includes completing the following tasks:

- Identify how and where documents will be acquired, such as scanning, faxes, multifunction devices, network folders, email messages, mobile devices, and imported documents.
- Identify the types of documents the application will process and the page types associated with each document type.
- Decide which data you want to capture from each page and which data might be manually typed or obtained by using database lookups.

- Determine which external systems, such as databases, could be used to provide index properties or validate data extracted.
- Specify the business rules that determine whether the captured data is valid.
- Determine how to handle documents that are structurally invalid, pages that are not recognized, data that does not meet the business rules, or characters that are not recognized with high confidence.
- Decide how you want to export the data and documents at the end of the workflow.

Before starting implementation, you must define the business requirements through collaboration with the various stakeholders. Initially this task involves examining the documents that you want to process, determining which fields you need to capture, and deciding what to do with the data and document after you capture it.

If you process various document types, you must decide whether the documents are presorted or processed as a mixed batch. If they are presorted, you might be able to simplify implementation by processing each type independently, with a separate application, workflow, or job for each type. However, if the documents are mixed batches, you need a more sophisticated system of page identification and document assembly.

Although the goal is to create a fully automated system, manual intervention is required at some points. The business requirements must specify how to determine whether the information is accurate and how to handle exceptions with the data or the process.

At the early stage of a deployed capture system, it is common to review documents even when they passed validation to ensure that the system is doing what is expected in a production environment. As time goes by and more confidence is built in the new capture system, the validation process can be reduced to review only the pages with exceptions.

One way to look at the design is to consider these three categories:

Document hierarchy	Defines the structure of the content that we process
Processing tasks	Performs the work of the capture system, such as scanning and identifying pages and recognizing data
Capture workflow	Sequences the tasks into processes that handle the needs of different functional areas or input channels

“Datacap v9.0 documentation” in IBM Knowledge Center provides extensive tutorials for each of these areas:

http://ibm.co/1GDLuiX

Be sure to read this guide before you design and implement any capture system.

4.2.1 Using FastDoc or Datacap Studio

Both FastDoc and Datacap Studio can be used to configure a Datacap application, so you might wonder which to use. Although there are fundamental differences between the two, the answer often comes down to preference. There is no right or wrong answer.

For example, a seasoned Datacap Studio user might prefer to stay in Studio, whereas those who are new to Datacap might prefer to start their configurations in FastDoc.

FastDoc provides a graphical interface for the workflow and compiled rulesets. This often makes it quicker and easier to get an application started. Although more complex applications might require you to use Datacap Studio to complete the configuration, FastDoc provides an easy and quick way to build your document hierarchy, create your fields, clean up images, set up page identification, build your workflow, test your configurations, and much more.

You can begin your application using FastDoc and enhance it as needed by using Datacap Studio. Enhancements might include creating custom rulesets or integrating custom actions.

4.2.2 Selecting the ideal template for your application

Starting the new Application Wizard prompts you to select either the Form or the Learning template. But not everything falls neatly into either of these categories. Often, an application might need a blend of the two. It is possible to select a template and add capabilities to the application as appropriate. For instance, you could start with the Forms template and add some of the Learning elements later as requirements dictate.

The Form template

The Form template is used to process structured images, such as account opening forms, tax forms, and other documents that contain a recognizable layout. Choose this template when you know the types of data that you want to capture and where the data is on each page type. Typically, the data on these forms is in a consistent location.

The Learning template

The Learning template should be used for unstructured documents. It is ideal in situations where you know the types of documents that you will process but the location of the data on the pages is unknown. For example, you might be processing an expense receipt and know which fields you need to extract but not know where these fields are located. The Learning template creates a workflow that enables you to add rules, such as “locate” rules, to dynamically find the data on the page. Datacap learns these new document formats when they are processed.

4.2.3 Document hierarchy

Datacap rules operate on batches, documents, pages, and fields. In Datacap, this structure is called the *Document Hierarchy* (DCO). The DCO is a core element of the design of the capture system. In addition to defining structure, the DCO provides the information that the system needs to assemble documents. It also enforces the integrity of the batches, documents, pages, and fields by using the information in the DCO. An application can have many DCOs to accommodate applications that require different classes of document structures.

The fictional bank in this book, Bank A, processes three document types: marketing postcards, loan applications, and bank statements. Within the configuration of the DCO, it is possible to define expected number of pages and the specific sequence. Within these document types, specific types of pages might occur only in a specific sequence. If the loan application has more than one page, the company can define the first page as a unique page type. This way, the system can determine where a document begins and enforce integrity. When the pages are reordered, a document is flagged as invalid if the wrong type of page is set to page one.

Beneath the batch level, the document hierarchy defines the following information:

- The document types that the application can process. You might have only one type, or you might have multiple types, such as the marketing postcards, the refinance loan application, and bank statement document types.
- The page types within each document type. Each document might have only one page type, or it might have multiple types. The loan application document type could include several pages: cover page, customer information page, signature page, and more.
- The number and order of pages within each document type. Pages can be required or optional.
- The data fields within each page type. Data fields can also be required or optional. The marketing postcard has different fields because it is a fairly basic document. It contains such things as name, phone number, and date. The loan application documents have many more data fields and might include address, social security number, bar codes, check boxes, signatures, and more.

In this scenario, after gathering information about the document types and their properties, we design a document hierarchy and enter it into the system by using Datacap Studio and FastDoc. For details, see Chapter 5, “Designing a Datacap solution” on page 113.

4.2.4 Capture processing tasks

In many instances, documents are acquired as a stream of pages where little information is known about the structure or content of the pages. Initially, the type of document and the processes that need to occur to correctly handle the document are unknown. For example, when documents are scanned, the input to the capture system might provide only a series of image files and the type of batch. The job of the capture system is to make sense of the images and perform a series of tasks that process them appropriately.

Typically, the tasks involved in the capture process include extracting useful data from the input, validating the input, formatting documents, and outputting the data and documents to business systems. Poorly scanned images and documents of various types that require human reviews might be mixed together. These factors introduce exceptions and variation that need to be detected and processed effectively.

The capture process performs the following essential tasks, which are incorporated into the capture system design almost always in the order shown:

1. Document acquisition

   Documents are input into the system by scanning, faxing, importing, mobile, MDF, email, or web services.

2. Image enhancement

   Images can be enhanced to improve recognition and readability and to reduce file size. This enhancement can be done at a scanner by using the built-in capabilities of the hardware or driver. Alternatively, it can be done by using the Datacap image enhancement features.

3. Page identification

   The type of each page must be identified (classified), automatically or manually. For example, a bar code can be used to automatically identify a page. A document often consists of a specific type of leading page followed by one or many trailing pages.

4. Document assembly

   The capture process assembles multiple images into documents where a single scanned batch or fax transmission can contain multiple documents. Information such as the page types, number of pages, and order of the pages provides the basis for automating document assembly. The document type is typically determined automatically by using the document creation function.

5. Recognition

   Recognition includes using optical character recognition (OCR), intelligent character recognition (ICR), optical mark recognition (OMR), bar code recognition, or database lookups to lift data and supplement the data with additional information.

6. Fingerprinting

   Fingerprinting is commonly used to differentiate between multiple formats of the same page type. Fingerprinting matches the best variation on a page type and captures the offset that is needed to adjust an image for locating data accurately.

7. Locating data

   Data in text on a page can be in zones or by using keyword searches through regular expressions.

8. Validation

   Extracting data by using any of the recognition methods has inherent limitations for many reasons. Examples of such reasons include a damaged source document, poorly scanned document, poorly printed document, and inaccurately entered data. Validation of the data is essential to obtain accurate results by using such techniques as check digits, length checks, format checks, cross-totaling calculations, value comparison, and data lookups.

9. Routing

   When exceptions occur, routing is used to queue batches or documents for exception handling. For example, a document that is missing a page or poorly captured might need to be fixed at a scanning workstation.

10. Verification

    Often a design goal of the capture system is to reduce or eliminate manual verification. However, when low confidence results or validation errors exist, they might need to be handled by human operators. Correct results need to be confirmed, and errors need to be resolved. Verification can include typing from recognition, typing from image when recognition is not used, and typing from documents when the documents are not digitized.

11. Export

    The system transfers documents and the data to external systems such as a content repository where they are stored and processed by the business. Extracted data is exported to XML files or databases to update applications.

Sequence of the design elements: Although most capture system design follows the previous order, the process can be done in multiple ways. For example, you can use fingerprinting (step 6 on page 92) as page identification (step 3 on page 91) before recognition (step 5 on page 92). Although fingerprinting is *not* a preferred way of page identification, it can be used.

Page identification: Multiple methods of page identification (PageID) are possible. Using fingerprinting as mentioned previously is one such method.

With Datacap, these elements are implemented as *rules*. Rules are run by the Datacap Rulerunner service. This method provides a flexible way to implement all of the variations and exceptions that are seen when capturing content in a scanned or document format.

In our scenario, many processing tasks and rules are already defined in Datacap. We merely need to adjust these tasks to meet the specific document structure. Datacap unifies the tasks definition with the document hierarchy. We configure rules and tasks with the same tool, Datacap Studio.

4.2.5 Capture workflow

During the data capture process, documents go through a workflow that consists of several tasks. Some tasks require operator intervention, where others run automatically. A workflow job consists of a series of tasks and defines a way to process a particular batch of documents. Because the tasks can be reused in multiple jobs, you can add as many jobs as you need to handle your processing scenarios. The design must include workflow jobs that specify and execute the capture process.

For example, in our book scenario, we might have several input channels, such as scan, fax, and email, for the same types of documents. We can construct three workflow jobs, one for each input channel, and have each job share tasks for recognition, data extraction, and export.

In addition to defining the process flow, the workflow also implements functional security so that you can determine who can access the work in progress and who can perform specific tasks with particular types of documents. For example, processing loan refinancing applications might be similar to handling new mortgage loan applications. However, the people who verify the documents might be in a different department. A separate workflow might be used to accommodate this difference.

4.2.6 Capture design considerations

This section highlights the areas to consider when designing a capture system and indicates the alternatives that are available. You can select from multiple options depending on your business and technical requirements.

Document acquisition

Datacap services various input channels that deliver documents in several formats. Channels or methods of capturing documents include scan, mobile device, multifunction devices, printers, fax, email, file import, and a web service. Some of the considerations for each channel are provided in the following sections.

Scanning

Direct scanning is typically done by internal users. Both desktop and web client options are available.

Desktop scanning is used in centralized scanning operations that use mid-range to high-volume scanners that have heavy-duty cycles. These types of scanners support continuous operation in multiple shifts. Even though a lower-cost scanner might have a high scan rate, it might not be designed for continuous operation.

The Datacap scanning user interface is production-oriented to support highly efficient operation of the scanner. In this environment, scanners are operated nonstop. Scan operators occasionally check the scan quality of images. The goal is to maximize the throughput of the production-level scanners.

Although multifunction devices (MFDs) can also be used to scan documents, they typically are not used for high volume environments but more for distributed capture environment. MFDs have integrated scanner, printer, copier, and fax capabilities. Production-level MFDs can be operated as stand-alone devices without being connected to a workstation. In this mode, the MFD control panel is used to control the scanner. Images can be transferred to a well-defined storage location by using the network filing, File Transfer Protocol (FTP), or email functions of the device. Datacap can import and process the documents by using its virtual scanning and email import actions.

A preferable MFD solution is to enable direct Datacap integration through the use of NSi Autostore or Imagine Solutions' Encapture products, which can be purchased through IBM. These products can integrate directly into the MFD console, which provides the ability to select a Datacap application to scan into, select document types, and enter index properties.

One area of common confusion is the difference between thin client scanning and operating an MFD directly. If you scan with an MFD by using thin client scanning, the MFD is connected to a workstation by using a TWAIN driver and the web user interface on the workstation provides the scanning control panel. This method is used with lower-end desktop MFDs and is not used with higher-end production MFDs.

Consider the following additional factors:

- Many current generation devices include image enhancement features that are run within the scanner hardware or in the scanner driver. In either case, the resulting image might have improved readability, improved recognition results, and reduced file size.
- Scanners are available for specialized purposes, such as remittance scanning and large format document scanning. These devices might not have Image and Scanner Interface Specification (ISIS) or TWAIN drivers. Therefore, they interface with Datacap by using import features.
- If you expect an MFD to be used full-time as a scanning device, consider using a dedicated scanner instead. Production scanning can handle larger scan jobs that might occupy an MFD that needs to be shared by a workgroup.

Web client scanning can also be used in centralized scanning operations but is more commonly deployed for distributed capture. Common deployment models are dedicated scanning stations connected to mid volume production scanners and user workstations connected to low volume scanners.

You can use Datacap Web or IBM Content Navigator. For organizations currently using or planning to deploy Content Navigator, it is recommended you use Content Navigator as the Datacap client. Using Content Navigator with Datacap can provide a single interface to users whether they are scanning or validating document, or browsing a content repository and provides numerous beneficial features such as detachable image windows for use on multi-window workstations.

Mobile

Adding support for mobile device can enable field workers to process documents in near real time. Currently available for both iOS and Android phones and tablets, Datacap Mobile acquires images and uploads them to a Datacap server for processing. Apple and Android phones and tablets support Content Navigator. Images can originate from the device's photo album or from the built-in camera.

As in the case with other clients, users need to log in to the application that they want to add documents to. Their credentials determine which application they can use.

Datacap Mobile dynamically detects document edges and automatically captures and rectifies the image only when quality criteria are met. This ensures that documents are of sufficient quality and can be processed downstream. Image enhancement tools are provided to help the user improve the quality of the image if necessary. Users can provide manual index values if necessary. When captured, the images are uploaded to Datacap for further processing.

IBM provides the Datacap Mobile SDK for iOS and Android to integrate document capture and image processing capabilities into custom applications.

Fax

Datacap software works with fax server products so that documents that are sent to a fax server can be imported into the capture system and processed in the same manner as scanned documents.

Fax is typically used by external users. The trend in many organizations is to reduce the internal use of fax for capturing documents. This trend is due to the lower quality of the image and the greater time needed to send a fax compared to remote scanning. However, because fax requires low bandwidth, its use is common in situations where only dial-up connections are feasible.

The primary disadvantage of fax is low image quality. The quality of the equipment varies resulting in inconsistent image quality. Fax image resolution is low. *Standard mode* provides a horizontal scan at 200 or 204 scan lines per inch. It provides a vertical scan at 100 or 98 scan lines per inch. *Fine mode* provides a horizontal scan at 200 or 204 scan lines per inch. It provides a vertical scan at 200 or 196 scan lines per inch.

Each fax transmission is received as a single TIFF or PDF file that contains multiple images. Datacap can burst the file into individual image pages for processing by the system. The image enhancement actions improve the ability of the system to recognize text. Datacap can normalize the dimensions of the image so that all the images are 200 dpi in both dimensions. It can also compress images to the TIFF Group 4 format.

Email

Datacap can capture and process email messages and their attached files. In addition to scanned images, Datacap can accept various electronic formats, such as word-processing documents and spreadsheets. Electronic documents can be converted to TIFF by Datacap so that they can be processed as images for data extraction and export.

Consider the following common scenarios for using email:

- Documents can be received directly from customers or other external parties. In the scenario in this book, customers who want to refinance their loans can be allowed to send supporting documents by email to a service email account.
- Email can be used as a replacement for fax as a way to transmit scanned or electronic documents.
- Email can be used to interface with MFDs.

File import

File import is a common method for inputting files into the system. The virtual scan (VScan or MVScan) features of Datacap are used to import files. File import can be done in an attended or unattended mode. In an attended mode, a user starts the virtual scan by using the desktop or web-client user interface. In an unattended mode, the virtual scan is run by the Rulerunner service, which runs as a Microsoft Windows service.

Consider the following common scenarios for using file import:

- Receiving images from an external party. For example, a financial institution might receive loan file images as part of the process for purchasing loans from another financial institution.
- Receiving images scanned by a scanning service. For example, large quantities of documents might be scanned by a third-party service as part of a backfile conversion.
- Interfacing with fax or MFDs.
- Interfacing with a scanner that does not have a TWAIN or ISIS driver. Some specialized scanners operate in this fashion.
- MVScan can use index files to process images. An index XML file is provided along with the images to import. Within this XML document, you can specify the document type, the properties to be passed, and the pointers to the images to be imported.
- Multiple MVScan threads can be configured within Rulerunner. They can point to the same or to different file locations. This is ideal for situations where you need a higher ingestion throughput.

Web service

Datacap displays the document processing capabilities as a web service. The web service can run the background document processing tasks. This method is used by software applications that need to process documents.

Consider the following common scenarios for using a web service:

- Processing previously scanned and stored documents that were not previously processed for recognition. A bank that stored loan documents when a loan was originated might want to perform data extraction on the same documents years later when a loan is modified.
- Providing a service where documents can be processed in an ad hoc manner. An organization might provide a service to upload documents for recognition and transformation through a web application or portal.
- When the Datacap web service can run in Microsoft IIS or as a Windows service.

Centralized capture

With centralized capture, dedicated staff and equipment process documents in a factory-like setting. Documents are mailed or delivered to the central location where documents are prepared into controlled batches. Batches are scanned on high-speed scanners. Other tasks, such as indexing, data entry, and fixup, are performed on separate workstations so that each task is optimized and labor and other resources are used efficiently at the central location.

Centralized capture offers the following advantages:

- Economy of scale
- Standardized processes
- Dedicated trained personnel who only do capture-related tasks
- Easier to maintain image quality controls
- Availability of original documents to verify authenticity

Centralized capture has the following disadvantage:

- Documents must be delivered to a central location.
- Users understand less about the documents.
- Corrections might require returning documents to the sender or interacting with remote users to correct problems.

Decentralized capture

With decentralized capture, remote offices or individuals scan, fax, and process documents, but they do not send the paper to a central location. Staff is not dedicated to performing capture activities. Capture might be done directly by the customer or by an external business partner.

Decentralized capture has the following advantages:

- Documents do not need to be mailed or shipped to a central location.
- Documents are stored into the repository more quickly.
- Users can correct errors immediately.
- Users understand the documents and can more accurately enter and correct data.
- Work can be offloaded to a partner or customer by using self-service.

Decentralized capture has the following disadvantages:

- Equipment is needed at each location.
- It is harder to maintain standardized processes.
- More users need to be trained.
- Users do not perform capture functions all the time and, therefore, do not handle the tasks as efficiently.
- Image quality varies, and image quality issues are more difficult to correct.
- Authenticity is more difficult to verify.

In many instances, organizations use a blend of these models. The capture system needs to accommodate the constraints and demands of the business. Organizations have multiple applications that require one or both models.

We must also consider the network capacity to determine whether it is sufficient to handle the required load. In some locations with low bandwidth, networks might need additional bandwidth to accommodate higher volumes of imaging network traffic.

In either scenario, the background processing of documents is handled centrally using the Rulerunner service. Background processing includes image enhancement, OCR or ICR, format conversion from input or for export, and export. Because these are processor-intensive activities, they are handled most effectively in servers or high-end workstations. In this manner, client workstations do not need to have software installed to perform these functions.

Image enhancement

Images can be enhanced to improve recognition and readability and to reduce file size. Image enhancement is most important when using OCR and ICR or to improve the format of faxed images. Datacap includes image enhancement capabilities for this purpose. The current generation of document scanners often includes image enhancement capabilities in the hardware or scan driver that can be configured in the scanning user interface. Use the capabilities of your scanning hardware for image enhancement, and supplement those capabilities with the Datacap enhancement features.

New in version 9 is the ability to change the order of execution of the image enhancement tasks, add or remove tasks, run tasks more than one time, and see changes to documents in real time. Also, several new image enhancement capabilities have been added.

Page ID and document assembly

Page ID and document assembly are often referred to as *classification*. This process identifies the type of each page in a batch and creates documents from the stream of pages. *Page identification* is the process of identifying the type of each page. *Document assembly* is the process of determining where each document begins and ends.

Orchestrated classification

Datacap performs automated classification by using the *orchestrated classification technique*. Orchestrated classification uses page identification rules, document integrity rules, and document creation rules to automate the classification process. Classification can also be done manually in a scanning or verification user interface.

Orchestrated classification uses a set of rules that takes a stream of pages. Then, it optionally enhances images, identifies each type of page using one of many methods, creates documents from the pages, and validates the resulting structure. All of the classification processing can occur in a single module in one workflow step. If necessary, you can have multiple types of classification modules. Classification can use any of the processing actions in Datacap.

Page identification

Documents are created and separated based on the page types and a set of document integrity rules. Pages can be determined by one of the following methods:

- Bar code
- Pattern match using image anchors
- Pattern match using text anchors
- Match image-based fingerprint
- Match text-based fingerprint
- Match regular expressions to recognized text
- Text analytics using IBM Content Classification
- Document structure using rules

Consider using bar codes as the primary method of page identification for forms that you control. When you do not control the layout of the form, you can use the other page identification methods depending on the characteristics of the pages.

Careful planning should go into selecting classification methods and the order in which they are used. Most applications will use several classification methods. For example, an application might first look for separator sheets, then page level bar codes, and finally text-based matching. Some methods are faster than others. For example, bar codes are faster than having to recognize an entire page looking for a specific keyword therefore, trying faster methods of classification first and working our way down from there is recommended.

Document assembly

In Datacap, the system determines document separation and document type by matching the document hierarchy to the identified pages. After pages are identified, Datacap uses the information in the document hierarchy to determine the correct document type. For example, a Loan Application page type is part of a Loan Application document, where a page type named Marketing Postcard is part of a Marketing Postcard document.

Each page has the following variables that define the structure of the parent document:

- Maximum number of pages of this type for each document (0 means no maximum)
- Minimum number of pages of this type for each document (0 means no minimum)
- Order, which is the position of this page relative to other pages in the same document (0 means any position)

Datacap uses the information in the document hierarchy to assemble individual pages into multipage documents.

A common approach to document separation is to use bar code sheets between each document or printing bar codes on the first page of a document. During scanning, several documents of varying lengths can be scanned in a batch, with bar-coded sheets separating each individual document. The system saves the documents as separate documents automatically. Barcoding can also be used to identify the type of document or pages.

Position bar codes vertically. Keep in mind that scanners and fax machines can produce vertical lines on a page when dirt is on the scanning sensor. If the line is parallel to the bar code line, it can make a bar code unreadable. If the line runs perpendicular across the bar code, it is readable.

Recognition, fingerprinting, and locating data

Recognition is used to read data from images by using OCR, ICR, OMR, or bar code technology. Recognition is used in three primary use cases: to automate document classification, to automate indexing, or to reduce data entry typing.

One of the methods of classification involves performing recognition on the document or a portion of a document looking for keywords, patterns, form numbers, or other meaningful information. Use recognition for classification carefully to ensure performance. For example, when looking for a form number that is always in the lower-right side, a well-designed application focuses recognition only in the area of the form. This enables the recognition process and the subsequent search for the form name to run much faster.

Indexing is the process of identifying the documents stored in the content repository. Documents are identified with properties that are stored in a content engine catalog. The process of entering these properties is called *indexing*. Users search for documents by using these properties. As a result, these properties must clearly identify each document with information, such as the name, social security number, and address. Usually, only a few properties are used to index a document.

Data entry is the process of typing data into a database or application system. Documents can contain dozens or hundreds of fields of data on many pages. In a manual process, users type data by looking at the paper document or at an image of the document in a window. Typing from a window is called *type from image*.

When we design the capture system in the scenario in this book, we use recognition features to read the data from images so that we can reduce the amount of manual typing.

Data recognition and extraction can be highly accurate when certain conditions are met. An understanding of the document characteristic is vital, because you need to choose the most appropriate techniques or combination of techniques that are most effective on the types of documents that you have.

Documents come in two classes: Documents that you control and documents that you do not control. Documents that you control are often internally generated documents that can be redesigned to make recognition more effective. If a document is not designed for recognition, it can be more difficult to process and can have lower confidence from the recognition process. However, documents that are outside of your control generally cannot be redesigned, and you must accommodate the existing format.

When you control the layout of your forms, a good practice is to redesign forms to improve recognition. In some cases, this redesign might be necessary to achieve high confidence recognition results. Some forms might need minor changes to improve results, but others might need extensive redesign. Engaging a form design expert is one way to achieve the best design for your documents.

The following factors can improve results:

- Use bar codes to identify document types and prepopulate a document with data. For example, internally generated documents can be printed with bar codes on them to identify the document type and indexing data. When they are returned, you automatically recognize the document type and indexing data rather than manually type the data. The business process capabilities have features that match the document to a pending task that is waiting for the document to arrive. Using bar codes in this way is included in our use case application scenario.
- Use machine print whenever possible. Forms that are completed and printed online generally contain printing that is easy to extract. Prefilled data of printed forms can also be easy to extract.
- Use hand printing only in controlled circumstances. Hand-printed information must be printed in boxes or other guides that show the user where to enter each individual character. Other types of handprinting require specialized software (describing it is beyond the scope of this publication).
- Using color drop-out during scanning when supported can also help remove handprint constraint boxes improving overall recognition.
- Clearly identify data locations by using unambiguous prompts or by using specific zones on the page.
- Include multiple fields that cross-check the data or use data that is designed to self-verify by using check digits.

Fingerprinting

Within a specific document type, often many variations can exist in the format and layout of the printed information about the page. The existence of variations does not refer to minor shifts in position on a page. Instead, it refers to the wider variations from a different version of a form or from documents that are created by outside parties where you cannot control the layout. In our design in the scenario, we must determine which method to use to handle these variations.

Fingerprinting is a technique that Datacap uses to differentiate between different multiple formats of the same page type. Fingerprinting matches the best variation on a page type and captures the offset needed to adjust an image to locate data accurately.

For highly structured documents, you can also use image or text-based anchor fields. These marks are on the page, in specific positions. This approach is effective for fixed-format forms where you have control of the forms design.

If you do not use fingerprinting or anchors, you can still deal with the format variation by using keyword searches and regular expressions to find data within the full text of a page.

In the design, we must examine all of the different types of documents and decide which approach is more effective.

> **Note:** The guideline is that if you can hold two different pages 10 feet away and see which page is different, you can use fingerprinting to identify the layout.

The Datacap Fingerprint Service should be used when you expect to have a large number of fingerprints. Besides using the fingerprint service, there is another helpful action that you can use to limit the scope of the fingerprints to those that are relevant: In applications that have multiple page types, you can use the `SetFilter_PageType` action from the Autodoc library. You can use this action to limit the scope to the page types that are relevant. For example, search only for marketing postcards, and ignore fingerprints of all other page types. Both of these strategies can help improve the performance of your Datacap application.

Location techniques

Data in text on a page can be in zones or by using keyword searches and regular expressions. These techniques can be used in combination to handle forms that have both fixed and variable data locations.

If you use fingerprints or anchors, you can accurately register the location of zoned fields. Zones can be prepared in advance or dynamically. The most flexible option is *Intellocate*, which enables Datacap to learn page layouts from users (see the next subsection for more information). It uses a hybrid approach that combines both zones and text searching.

When we design a system, we must examine the individual documents to determine which location technique can be used for the data on our documents

Intellocate

Intellocate is technology that enables Datacap applications to learn. Location rules are used to automatically locate some of the data from these documents by using keyword searches or regular expressions. Information that cannot be automatically found by using Intellocate can be identified and captured quickly and easily by a verification operator by using the Click N Key capability.

With Click N Key, the operator clicks the words on the image, and the data is entered into the data field. Behind the scenes, the system remembers the locations where the user clicked. When this task is complete, Intellocate saves the zones for the fingerprint. Then, the next time that a similar document is encountered, the fingerprint is matched, and all of the data is read by using the zones.

Data validation

The purpose of validation is to determine whether captured data conforms to specified business rules, as in the following examples:

- Does the loan amount lie within permitted limits?
- Are dates valid and within a permitted range?
- Has the form been signed?

Datacap performs validation by using rules that you create and attach to specific items in the document hierarchy. For example, to check whether the refinance amount lies within permitted limits, you might first create a rule that performs the following tasks:

- Ensure that the amount field contains numeric data in a valid currency format
- Determine whether the value is less than or equal to the maximum permitted limit
- Perform exception handling if the value is invalid or higher than the permitted limit

OCR and ICR read what the user entered. The user can still write or print invalid data on the document. The scope of the capture process usually includes validating that data was correctly read from the page and that the content of the data is valid.

Validation can include simple field-level checks, field cross-checking, and lookups to external data sources. At the field level, it checks the ranges of values, valid format (for example, dates), choice lists, valid currency amounts, and so on. Cross-checking can include totaling columns and checking against total amounts. Datacap can query database tables to look up valid values. Lookups are used to check account numbers, product codes, and other sorts of master data.

Depending on the business use of the data, the application might need absolute data accuracy. But sometimes, you might want applications to accept lower-confidence data.

For example, if you are processing loan transaction, you want data to be accurate. However, if you are processing survey cards, you might want to accept incomplete responses of lower confidence because you are more interested in the receiving as many responses as possible.

For each type of page and for each field, you must determine the level of confidence that is flagged by the system and displayed to an operator for verification.

Validation is run in the background task after recognition and at the Verify task. Data that does not conform to business rules is flagged. Documents that do not conform to business rules are routed for verification by using workflow. Data can be flagged down to the character level. Validation is also executed from the verification user interface when a user types corrections.

Routing

Workflow routes exceptions for manual verification. You can route an entire batch, or you can split batches so that problem documents are handled in a separate batch from the good documents.

In the loan refinance scenario, we must determine the types of exceptions that we will handle and who will handle them. Common types of exceptions include rescanning, page identification, incomplete documents, data verification, and data exceptions.

Verification

During verification, an operator views data entry panels and document pages for manual checking, for possible correction, and to type data. Display pages to an operator when one of the following primary conditions exists:

- The batch failed document integrity checking during document assembly.
- A page contains one or more characters or OMR fields that were marked “low confidence” by the recognition engine.
- A validation rule failed, indicating that the data does not conform to business rules.
- The application does not recognize that data and verification windows are being used to type data from the image or the document, which can be for indexing or data entry purposes.

When a batch fails document integrity checking during document assembly, you can have a user manually identify pages, by using a special verification task called *Flex ID*. Flex ID handles the manual page identification, and you display the thumbnail images of the page. Then, the user rearranges the thumbnails and selects the page type for unidentified pages.

The other conditions require the user to enter or correct data. Several thin- and thick-client verification user interface options are available. All of these options display the image, the data fields, and snippets of images where data is on the image page.

Take a single-pass approach to verification. Some other systems promote a two-pass approach where individual character-level corrections are handled in the first pass, and in a second pass, field-level corrections are made. Our experience is that a single-pass approach is more efficient. The user interface has keyboard shortcuts that navigate efficiently at the character level, making a separate first pass unnecessary.

You can control what is displayed to the user in the design and configuration. As part of the design in this scenario, we decide what level of verification is needed. Many options are available. For example, we can display every document and page, only pages where we have data, the first page of a document, only documents and pages with exceptions, and pages that do not conform to business rules. This setting is a business decision and varies depending on factors such as the types of documents, business controls, or the comfort level of the user with automating the process.

Multipass verification

To satisfy business requirements, you can consider whether you want more than one person to verify the data. Multipass verification can display the same page to multiple operators to ensure accurate data entry and verification. In some cases, the business financial controls require a separation of duties that requires more than one user to enter or validate specific data fields.

Datacap supports two main implementations of multipass verification: two-pass and double-blind. Other implementations are possible, but this book focuses on these two, which are supported by the standard user interfaces.

In two-pass verification, the following process occurs:

1. An operator (or a recognition engine) enters the initial value for each field.
2. Datacap displays the page to a second operator but hides the initial values. The operator enters a new value for each field. If you are using a recognition engine to implement the first pass, you might choose to show only low confidence fields to the operator.
3. For each field, Datacap compares the new value to the initial value. If the values match, Datacap accepts the value. Otherwise, the operator must re-enter the value. Datacap accepts the value only after the operator enters the same value two times consecutively.

In double-blind verification, the following process occurs:

1. An operator (or a recognition engine) enters the initial data values.
2. Datacap displays the page to a second operator but hides the initial values. The operator enters a new value for each field, and Datacap saves all of the values (no comparing).
3. Datacap displays the page to a third operator. The operator can see both the initial value and the second value.
4. For fields where the initial value and the second value are different, the operator must select which value is correct or enter a new value. If entering a new value, the operator must enter the same value two times, consecutively.

Web-based clients

Datacap includes several different user interfaces that have different design features.

Table 4-1 summarizes the key functions of various Datacap web interfaces.

Table 4-1 Functions of various user interfaces

Function	Web page
Key-from image, manual page identification, manual registration (anchor clicking), two passes, and double blind verification.	`aindex.aspx`
Verification interface, custom panels, and line item details. Line items can be navigated using arrow buttons	`averify.aspx`
Verify recognition and overlay data entry fields on the image, used by MClaims (Medical Claims), specifically	`imgEnter.aspx`
Modern virtual scan interface, supports mixed types, (probably) does not burst multi-page TIFFs	`pickup.aspx`
Manual page identification with image thumbnails (for example, Web FlexID), typically not designed for large batches	`ProtoID.aspx`
Modern fixup panel, (probably) supports large batches	`restruct.aspx`
Basic remote scanning page	`scancl.aspx`
Upload task, sends locally scanned files to server	`uplbfcl.aspx`
Modern verification interface (successor to `APTLayout/prelayout.aspx`), line items displayed at once	`verifine.aspx`
Virtual scanning of files, does not support mixed types, automatically bursts multi-page TIFFs	`vscancl.aspx`

New with version 9 is the integration into Content Navigator. The addition of Content Navigator to the supported web platforms allows for more flexibility in how the users can use Datacap. Through this single interface, users can scan, classify, index and verify, create custom desktops and layouts, view the job monitor, perform Datacap administrative functions, and access any connected repositories.

Organizations currently using Content Navigator or with plans to roll out Content Navigator should consider using it for Datacap also, because it provides an enhanced user experience that is consistent with other ECM products that they use.

Content Navigator is easier to configure, offers flexibility in presentation layout, allows for the detaching of the view panel for customers who might be using multiple monitors, and more.

Fixup tasks

Fixup activities adjust and enhance pages, move pages within the batch, reconstitute documents, and reorganize the batch.

If a workstation has a scanner attached, the fixup can also rescan. *Rescan* is a physical process that scans one or more pages and replaces existing image files with the new files. One constraint for rescanning is that the workstation where rescanning occurs must have access to the physical documents.

In centralized operations, scanned documents are often stored in boxes onsite for a short time when rescanning occurs. In decentralized environments, batches must be routed to the person who scanned the document.

Many customers find it is more efficient to rescan an entire batch rather than to pull out the individual document and rescanning the individual page. This preference depends on the application.

Sometimes batches are sent to a fixup task to delete documents from a batch. For example, you might need to send the original document back to the sender if it is not a valid document. In this case, an operator must pull the original paper document from the physical batch and send it (perhaps by mail) to the originator. The images can be flagged for deletion from the batch.

Export

Datacap supports data and document export.

Data export

Datacap can export data to a text file, an XML file, or a database. This choice depends on the means of interfaces that are available in the target application. These three methods are equally easy to configure. You can export multiple formats for the same batch.

Document export

Document export creates documents from the scanned pages and stores the documents in one or more systems.

In addition, the Datacap software can export to a file system, other IBM enterprise content management (ECM) systems, third-party ECM systems, collaborative systems including Microsoft SharePoint and CMIS-compliant ECM systems. When a batch is exported, the destination for each document is determined at the document level. Therefore, documents in the same batch can be stored in multiple systems. It is possible to export to any number and combination of repositories, databases, and files.

Other considerations when exporting the documents to IBM FileNet Content Manager, you can export of multipage documents as multiple content elements. You can also upload any file type to the repository and set attributes for the destination folder and its subfolders. You can also add new pages to, replace pages in, and delete pages from an existing document in the IBM FileNet Content Manager repository.

In version 9, Datacap has expanded the export capabilities of the IBM Content Manager repository export by adding nine new actions. The following are some of the key enhancements now supported by IBM Content Manager:

- The creation of child items
- Searching for existing items in a Content Manager repository
- Adding, deleting, replacing pages of a document already in Content Manager

When designing the system, you must determine the output system, the file format, and the document properties of the exported documents.

Considerations for exporting file formats

The primary output options are TIFF, PDF, and PDF/A. Documents are processed as individual TIFF image pages, but they can be converted to different formats for export. For example, scanned or faxed images in TIFF format can be converted to PDF/A for export.

In addition, consider the following export options:

- Documents can be exported in original format. For example, when a customer sends a Microsoft Word document by email, it is processed by using the image functions as TIFF pages. When the export is done, the original Word document can be exported.

- Documents can be exported in multiple formats. In the same example, the Word document can also be rendered in PDF/A format and stored for archival purposes. It can also be stored in the original Word format.
- PDF documents can be image only or made text searchable. Creating a text searchable PDF is ideal for situations where the destination repository is content search enabled. This enables users to search for content using index properties and words from the document.
- Images can be redacted, where specific area of the image is erased or obscured. Redaction can be used to cover ID numbers, credit card numbers, or other protected information.
- Additional capabilities are available for conversion to other formats including a rich text format (RTF), HTML, Microsoft Excel, Word, and various text formats. These formats are used for specialized applications and are secondary options for imaging applications.

Considerations for exporting document properties

When documents are stored in the repository, you store the document and catalog the document by using a small set of document properties, such as the name, address, social security number. You use the data collected throughout the capture process on the batches, documents, and pages. You store selected data fields with the documents into the content repository.

For example, in FileNet Content Manager, each document belongs to a document class where the document class specifies a list of document properties. These classes are mapped to corresponding Datacap document types, in a one-to-one or one-to-many relationship. The properties of the document class map to Datacap fields and variables on the batches, documents, or pages.

A unique feature of Datacap is the ability to update the properties of an existing document in the FileNet system, not just the document you are exporting. For example, the document you are exporting is a change-of-address request, and a field contains an updated postal code. In this case, you can update the postal code on other documents that are already stored in the FileNet system.

You can also use this update feature to implement an early export scenario. In this case, documents are exported before the entire recognition and verification workflow job is completed. With this scenario, documents are exported to be stored under document management system, even though all the steps of the workflow have not completed. In a later workflow task, additional data can update the document properties.

Exporting to IBM Case Manager

After a document is stored in the FileNet Content Manager repository, both the document and the data are available to a Case Manager process. The act of adding the document triggers an event to, initiating a new Case process or updating a currently running Case process. Any data in the document properties, and the document, can be included in the Case process.

With existing Case processes where the process steps include data entry, consider shifting the data entry function to the capture system. With the data recognition and extractions capabilities of the Datacap, these functions can be implemented so that they require less manual typing of the data.

4.2.7 Discovering the capture process

Every organization has a different starting point with different requirements. Some organizations already have implemented a capture process and are updating their system to take advantage of advanced capabilities. Others are implementing capture for the first time or are using Datacap for the first time on a new application. Some users want to scan documents and store them in an ECM repository. However, others want to extract data from documents for updating business systems. Yet others want to focus on classifying documents and reducing manual paper document preparation. In any case, understanding the capture process is a key step in the design process.

As a starting point, in the refinance loan application scenario, we must identify the business requirements through collaboration with the various stakeholders. The process of gathering required information about a business problem is called a *walkthrough*. In a walkthrough, you learn about the current sources and methods of handling documents, and examine the documents. You learn the characteristic of the documents, how they need to be processed and stored, and determine which fields need to be captured and what to do with the data after you capture it.

In addition to the mechanics of configuring the capture application, you must consider the physical aspects of the paper handling tasks, such as whether document types are mixed together or presorted. If they are presorted, you might be able to simplify implementation by processing each type independently. If you process mixed batches, you can automate and reduce the amount of manual sorting using the orchestrated classification techniques.

If data or index values are manually typed, you can examine the characteristic of the data printed or written on the documents and determine whether the data can be extracted by the software. Alternatively, you can propose a redesign of the document layout or forms so that the data can be captured more effectively.

Although the goal is to create a fully automated system, at certain points manual intervention is required. The business requirements must specify how to determine whether the information is accurate and what you will do if a problem arises.

Because this book is an introductory guide, it does not provide a detailed methodology for determining business requirements. Instead, it provides guidance about the key information that you need to gather and review.

4.3 Gather requirements

This section lists the questions that help to identify the application requirements and the relevant details to design the capture system. The information that you discover includes the following categories:

- Current capture or document processing environment
- Physical locations that receive and process documents
- Types of documents, their characteristics, and the data that they contain
- Business rules that validate whether the data is valid
- Document volumes and time constraints
- Business requirements for dealing with exceptions
- Output requirements for data and documents
- Ingestion mechanisms and their requirements
- Hardware and software requirements

4.3.1 Requirements for current capture or document processing environment

With this category, you discover the characteristics and details of the business processes and systems that are currently in place. You look for the specific tasks that are performed, the sequence of those tasks, and the overall time it takes from document arrival through completion.

Ingestion

Documents can originate from numerous sources including: scanner, MFD, fax, email, file share, mobile, web upload, external web service call. The ingestion process must evaluate these sources of content and determine which of them need to be included in the configuration of the solution. Each source of document can be processed independently if necessary. In multiple source ingestion scenarios, it is common to have different requirements for different sources. For example, emailed documents might require conversion to TIFF, whereas scanned documents would not. Each source can have specific tasks associated.

Scanning

If the current process involves scanning documents, you must identify the current systems and methods. You can consider redesign of the existing processes in light of the capabilities of the enterprise imaging system compared to existing methods. You can evaluate potential reuse of existing processes, equipment, and systems.

Identify the scanning requirements:

- Are paper documents currently being scanned?
- At what point in the business process are they scanned: upon arrival, in the middle of the process, at the end of the process, or a mixture?
- What equipment and software are being used to scan the documents?
- Will the current equipment be replaced, or will it be used with the new system?
- Can the scanners handle the projected peak volumes based on comparing the scanner specifications to the scan volume?
- Will the scanner handle deskewing and noise removal?
- For each location, will scanning be done by using thick-client ISIS or thin-client TWAIN scanner drivers?

  The preferred practice is to test a specific scanner interface, driver, and scan hardware in a test environment.
- What happens to the paper documents after they are processed? Are they stored onsite, returned, stored off-site, or destroyed?
- Will the new system change the way paper documents are handled after they are scanned?

Processing

Processing paper documents is a labor-intensive operation. By explicitly documenting the current processes, you can identify the specific areas of process improvement.

Identify the processing requirements:

- How many people “touch” the document from arrival to completion, and in which departments or locations do these people work?
- What is the current document handling process?

- Are documents processed centrally or at remote locations?
- How many people are involved in processing documents?
- Which processing is currently being performed?
  - Receiving documents, logging, counting, batching, and date-stamping
  - Sorting documents for filing and distribution
  - Preparing file folders
  - Filing documents
  - Distributing files or documents for processing
  - Photocopying for distribution
  - Manual typing of data
  - Retrieving files from file cabinets
  - Searching through files to find documents
  - Matching documents against exceptions reports
  - Re-filing documents and files
  - Pending or suspense file management
  - Keeping calendars or diaries to track follow-up documents
  - Searching for misplaced or lost files
  - Reconstructing lost files
  - Purging files and removing selected documents for disposition
  - Transporting documents to and from storage rooms or off-site storage
  - Filing internal forms or copies of correspondence

Policies and systems currently in place

Identify the policies and systems that are currently in place:

- Has our organization approved the destruction of original paper documents following scanning?
- What systems are used for tracking and inventory of paper documents and files?
- What ECM or other systems are currently involved in the current scanning or capture operation?

Time frames

Identify the requirements regarding time constraints:

- How long does it take for a document to be processed from arrival to completion?
- Are there significant differences in time depending on document type? If yes, identify the differences.
- What steps in the process take longer than you expected?

4.3.2 Processing location requirements

This section identifies where documents originate and how to gather them for processing.

Physical documents

Identify the requirements for physical documents:

- How many physical locations create or receive physical documents?
- Are the physical documents processed in the location where they are received, or are they moved to a central location for processing?
  - Are they moved by mail, internal courier, or external courier?
  - Are photo or scanned copies made before they are moved?

Electronic documents

Identify the requirements for electronic documents:

- How many physical locations create or receive electronic documents?
- Are the electronic documents processed in the location where they are received, or are they moved to a central location for processing?
  - How are they moved: By email, electronic media, file copying, or file transfer?
  - Are copies made before they are moved?

4.3.3 Document type requirements

The questions in this section help to identify the documents types, how they are created, and their characteristics. You must identify and gather single and multiple page samples of all document types.

Identify the requirements for document type:

- What are the document types and any subtypes, that we process? Consider the following examples:
  - Packing slips for complete, partial, back ordered shipments
  - Invoices, including purchase order invoices, non-purchase order invoices, preapproved invoices, trade invoices, non-trade invoices, and credit memos
  - Attachments, including shipping confirmation notices and acknowledgment of receipt forms
  - Loan applications, including the application form type by form number
  - Insurance claim, such as the claim form by form number
  - Tax forms, including the form number and year
- Who creates the documents?
- Can the design of the documents be changed if necessary to increase recognition accuracy?
- If documents are created by external parties, approximately how many sources are involved?
- What is the input source for each type of document: scanner, fax, email, or other systems?
- For each type of document, does it have a fixed number of pages or a variable number of pages?
- What is the number of pages per document?
- For images, what is the image resolution and format (black and white, color, gray scale)?
- What is the input file format for electronic documents?
- Do documents contain more than one business transaction?
- Do people stamp, mark up, or write on documents as they are processed?

4.3.4 Captured data requirements

Whether you use recognition technology or manually type the data, you must identify the characteristics and processing details of the data on your documents. With this information, you can determine the data recognition requirements and other aspects of handling the data, including validations, lookups, verification, indexing, and data entry.

Identify the requirements for captured data:

- What fields should be manually entered at the batch level (for example, Scan Date, Expected Number of Documents, or Expected Number of Pages)?
- What fields should be captured at the document level (for example, name, address, social security number, phone numbers, check box, signatures)?
- For each document type, is data primarily machine printed or hand printed?
- For hand printed documents, is the print constrained or unconstrained?
- Are there pages that do not have data that must be recognized, such as attachment pages? It is common for forms to have instruction pages that are scanned but that do not have data on them.
- How is data located on the pages where you need to use recognition to read the data?
  - Fixed form layout. Fields are on specific zones where the location can be used to find the data.
  - Variable form layout. Fields have text labels where a search for the text label can locate the field.
  - Data is contained in a bar code.
- Is data validated by using an external database?
- What are the business rules for validating the values of the fields?
- Do fields have lists of valid values?
- Is it data optional or required?
- Does the data printed on the page conform to a repeatable pattern? (For example, Loan Application Number starts with the letters LA followed by six numerics, a hyphen, and three numerics.)

4.3.5 Verification requirements

Verification intersects users with the documents. You must understand where these users are located and what tasks they are authorized to perform on each type of document. Business rules need to be applied that might mirror existing practices for handling paper-based data entry. Verification might also be desirable as a quality control step to ensure that every image is readable.

Identify the requirements for verification:

- Will verification be handled in a central location or from remote locations?
- Are there business rules or policies that will require multiple verification steps?
- Who will perform verification?
- Does verification need to restrict access to specific document types by different groups of users?
- Do we need to display every document or page or can we display only documents or pages where we have exceptions?

- Will some documents require manual page identification by an operator?
- Based on the information gathered on input documents, captured data, and export requirements, how should low confidence data, invalid data, unidentified documents, and incorrectly identified pages be handled?
- When recognition results are high confidence, do you want an operator to view the document anyway?
- Do operators need to visit all fields with low confidence characters?
- Under which circumstances can the operator split out a document from the batch to finish processing the other valid documents in the batch? How should the split-out documents be handled?
- Should operators be able to mark document for deletion (documents will not be exported)?
- Should deletion trigger a follow-up process or automatic notification?

5

Designing a Datacap solution

This chapter gives some direction and tips for the initial design of a Datacap application. A complete design that takes in all of the factors is essential to creating a quality application. As further design details are found during the development process, they should be noted in the design document. Then, at the end of the project, there will be a document that is invaluable for future enhancement and maintenance of the application.

This chapter covers the following topics:

- Start at the end
- Obtain sample input
- Choose your starting point
- Configure external processes

5.1 Start at the end

Every computer application takes some input, processes that input, and produces some output. In designing a Datacap application, it helps to keep this in mind. In most cases, the input is in the form of an image, Datacap extracts data from that image, and the image and any associated data that is extracted, or captured, from the image is exported to one or more content management repositories.

During data extraction processing, additional data not present on the image can be obtained from several other sources, such as databases, web services, or operator input.

Start image data extraction by compiling a complete list of all data (fields) required by the system or systems that Datacap will export to. To make this process easier, you can use the Application Wizard.

Select **Create a new CMIS-based application** to create fields in your new skeleton application automatically, based on a CMIS-compliant repository. If Datacap will export to other repositories, you must create the fields by using FastDoc or Datacap Studio. Choose the top option, **Create a new RRS application** (Figure 5-1).

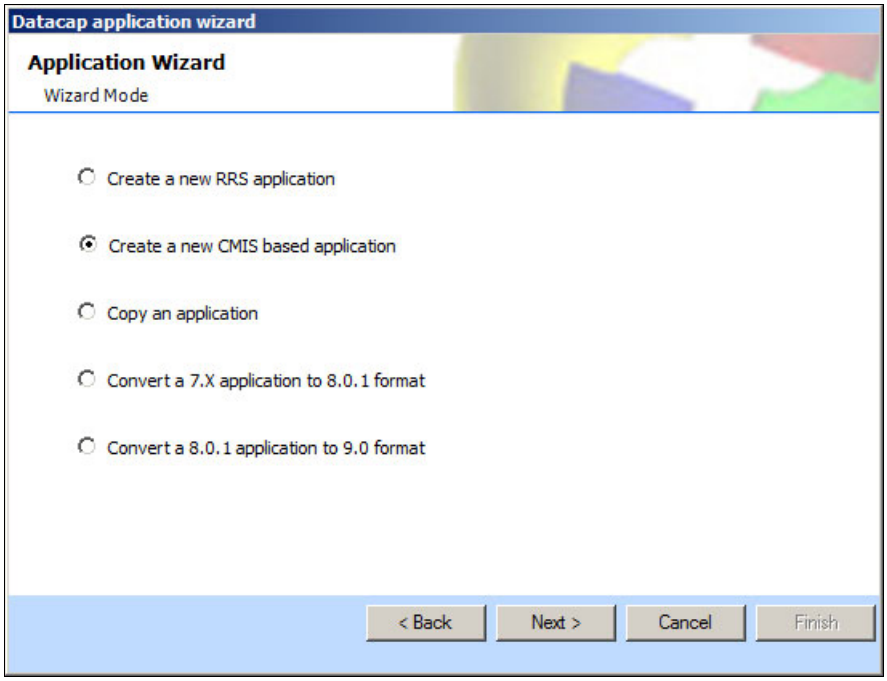


Figure 5-1 Selecting an application in the Application Wizard

Consider the data types, field lengths, and other data-level restrictions to that you need to adhere to successfully export to the target system. Also, make a note of which fields are required and which ones might be blank. Then, determine how to handle exceptions, such as what happens if a date is required for the export but an invalid date or perhaps no date at all captured from the image. Knowing where you need to end up is critical for some of the design decisions you make early on.

For every item you expect to export, it is critical that you answer the following questions early in the design phase:

- In what field name or variable am I going to store this data?
- Where will I get this data from?
- How will I get the data?
- How will I validate the data?
- What do I do if the data is not available to me?

5.2 Obtain sample input

The next step in designing your capture solution is to understand the input you are working with. In addition to the devices and input methods that the application will support, you need a representative set of sample images from each source. It is important to make sure that any sample images are truly representative of what the input will look like when the capture solution is in place. In many cases, when you request sample documents, you are provided with samples after data entry was done on them. Such samples often contain markup,s such as circles, check marks, and other notes made by data entry operators when they carry out manual data entry from paper. Insist that the samples represent the quality of images that you will receive when your application goes into production. Some coaching on the resolution, compression format, image type, and so on might be necessary to receive images that you will want to work with.

Without any guidance, most customers tend to provide either their best examples or the worst. You need some of both and quite a few in between.

The skeleton applications created using the Application Wizard pull images from subfolders in the imaging directory of your application. There is one folder for single-page TIFFs (to simulate a stream coming in from a scanner or fax) and another folder that accepts multipage images in a variety of formats (to simulate what you might get from an email or MFD).

Divide the images that your application is to process into several test sets and you can place them, in folders, inside or alongside of the predefined input folders. This way, you can copy easily new samples into the input folder when you need to test specific parts of your application.

Note: It is not considered good practice to develop an application while dropping pages into a scanner each time that you want to test a code change. Instead, during development, work with images read from disk. This saves time and enables you to faithfully replicate the same processing over and over when you discover and correct issues in your application. Similarly, use enough images in your development runs to cover the breadth of features in your application, but not so many that test runs take too long to test a change.

With the skeleton applications already set up to handle a variety of common image types, you can plan how to get from here (input) to there (output). Before going to production with your application, remember to configure your virtual scan job to delete or move the images outside of the input folder after they are ingested.

5.3 Choose your starting point

In most cases, you will use the Application Wizard to start your application (Figure 5-2). At the time of writing, there are two starting templates available when you create new application using the wizard:

- Forms
- Learning

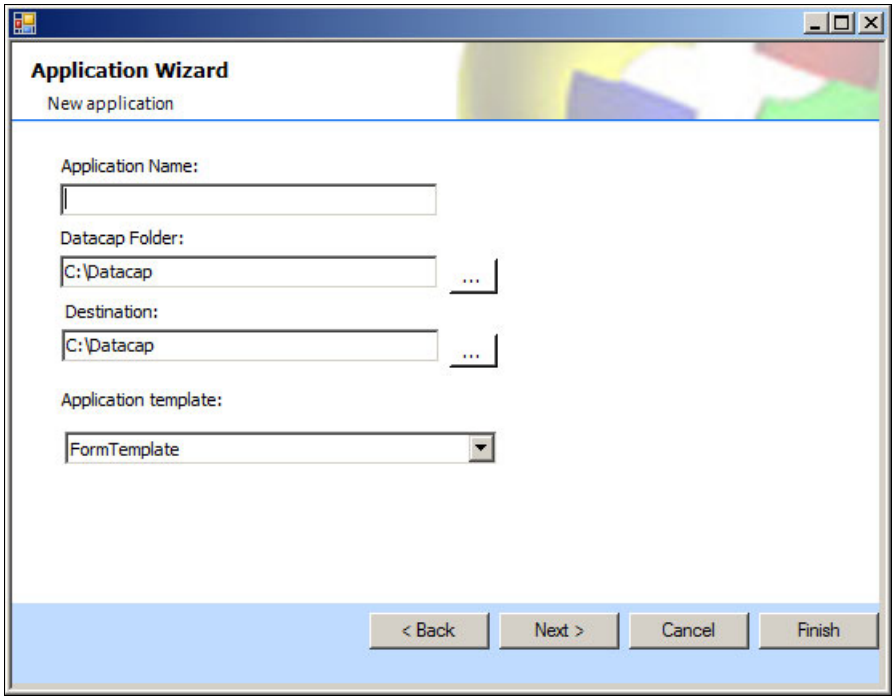


Figure 5-2 Datacap Application Wizard Form template

However, you are not limited to these two starting points. You can copy and alter the Forms or Learning templates to have templates of your own, complete with your commonly used rules, rulesets, and action libraries that you have created and use often. Then, that can become your new starting point if you choose. Simply put your new template in the `Datacap/Templates` folder and they will appear in the drop-down menu along with the Form template and Learning template.

You can also start with an existing application that closely matches your use case. That might be one of the Foundation Applications (APT or mClaims, which might be separately licensed) or with some other application that you are familiar with.

To start with an existing application, select the **Copy an application** option in the Datacap Application Wizard (Figure 5-3).

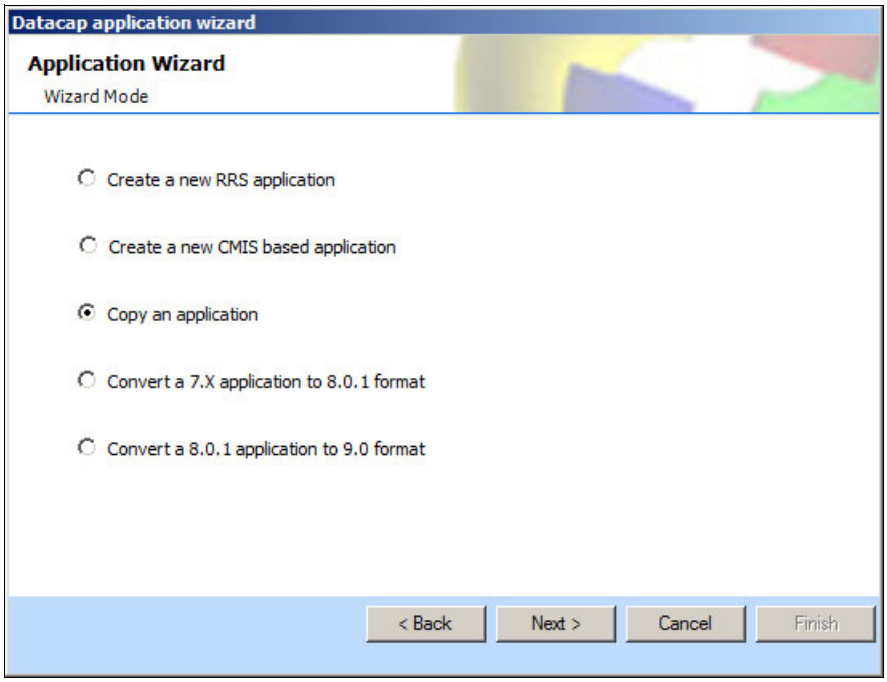


Figure 5-3 Datacap Application Wizard, Copy an application selected

Note: Do not modify the APT, mClaims, or template applications directly. Instead, create a new application based on them, with a different name, and then change those new applications. If you modify the templates or a foundation application, an update or reinstall of the Datacap software will overwrite your changes to those applications and you will lose work. For example, if you use the Application Wizard to copy APT to MyAPT or use the Learning template to create an application called MyNewApp, your new applications are safe from being overwritten on an install or upgrade.

The Forms template is used to design an application that recognizes different areas on a form in a specific manner. Use this template when fields are found in the same position, on the same page of the document, every time. The Forms template relies on field-by-field, rather than full page, recognition. With field-by-field recognition, you can give the recognition engine different parameters for specific fields. For instance, you can configure one field on the form to recognize digits (0-9) only. You can also specify that it is to recognize with hand print, MICR, OCR-A, machine print, dot matrix print, and so on. Defining field length and even supplying a dictionary is possible with field by field recognition.

With Form applications, fields are normally associated with the specific pages that they are on, recognizing the name from of Page1, but also recognizing an amount from page 2. Documents structured in this way always have the data in a known location, at a specific place on a specific page.

See Figure 5-4. With structured forms, normally you have a different Document Hierarchy (DCO) type for every page, and each page has child fields to hold the data from that page.

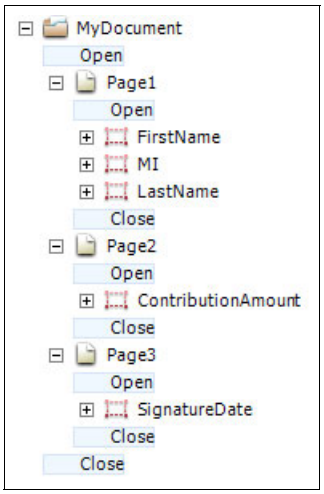


Figure 5-4 Organizing document by page

Because you know what every image coming into the system should look like, the Forms template detects early on when an image does not look like expected input. In those cases, the batch is routed to a Fixup operator so that the problem images can be identified, oriented, rescanned, or otherwise corrected so that processing of the batch can proceed.

Different from the Forms template, the Learning template is set to handle unstructured or semi-structured documents, such as invoices, proofs of deliveries, and correspondence. That is, anything where you have no control over how many variations of these documents your application will need to process. Compared to the Forms template, the Learning template handles documents in a more general way.

For example, in a Forms application, it is easy to identify a document using a bar code or fingerprinting, so document structure is apparent when the pages are identified. In a Learning application, we look to the image stream to tell us where documents begin and end. As the default configuration, the Learning template treats all images at the start of a batch as single page documents until a document separator sheet is encountered in the batch. From that point on, images between the document separator sheets are considered to be in the same document.

It also handles multipage image input by making the assumption that each file represents one document, whether it is one page or multiple pages stored in the file. For example, a three-page PDF file will be converted to single page TIFFs, with all three pages in the same document.

Another difference is where data is stored in the DCO hierarchy. Because the input is not structured, we do not know on which page of the document we are likely to find the data we want. For this reason, in the Learning template we set the first page of the document to Main_Page and the all other pages to `Trailing_Page`. We define all fields for the document on the `Main_Page`, even though the data might actually be found on the second or third page of some documents, and the first page on others. Trailing pages might have the data on them, but we typically store it all on the first page because there is no set structure to know which page the data was actually found on.

Figure 5-5 shows the document hierarchy of APT, the standard accounts payable application. All data is stored on `Main_Page`, and `Trailing_Page` has no fields at all. However, the data can come from any page in the document.

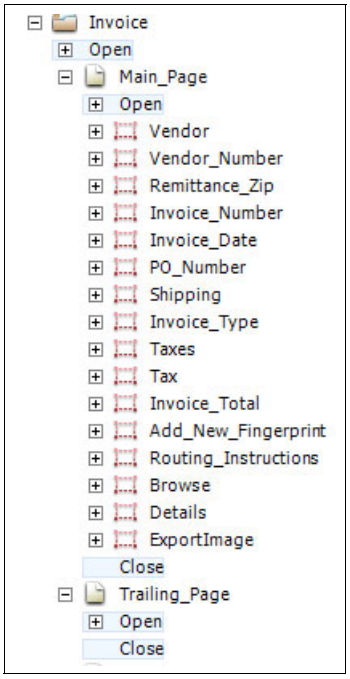


Figure 5-5 Document hierarchy of APT

Because we do not know where data will be located on the page, we cannot recognize different zones of the using different recognition parameters. Instead, the Learning template recognizes the entire document with a full-page optical character recognition (OCR) (machine print) read. It then builds a connected component collection (CCO) file that contains all of the data from the recognition results, which we can then search for the data that we want regardless of which page it is actually on.

The “learning” part of the Learning template comes from the fact that after verification, when all data was successfully found, we know the locations for the data for this particular type of document in case we ever encounter the same document (with different data) in the future. This feature is called Intellocate and is used in the APT and Flex applications. The Learning template essentially uses the same input and learning process as those two applications.

5.3.1 Analyze the images (image enhancement)

Any image that is to have data extracted from it needs to go through an image enhancement process to make extraction more reliable. At the least, images should be rotated properly and deskewed before using them as the source of your data. Lines that are close to areas that you need to extract from should be removed and the image should be de-speckled also.

Although it is possible to enhance each page individually, typically enhancement is required for all pages in a batch. So in practice, the same settings are applied at batch level. If you work with bar-coded images, make sure the minimum line length is long enough not to affect the bar code, or add a ruleset before calling image enhancement that recognizes the bar code.

Bar code recognition is somewhat tolerant to noise, orientation, and skew so recognition before any image processing at all is usually sufficient. If you work with images containing hand printed characters in a segmented field (boxes or combs), make sure that these markings are removed completely, either by the scanner using a color filter or by your image enhancement settings.

5.3.2 Analyze the image stream to identify pages (page identification)

You must determine how images come into the application. Images come into your system in a certain order, as a stream of images, or image stream.

An application might need to handle different image streams simultaneously. For instance, when getting images from a scanner, you might have bar-coded separator sheets that can be used to indicate where document separation occurs. The same application might also obtain input from a fax or email server where separator sheets are not used. Instead, the source image structure in the image stream is used to determine document separation. If there is no other way to identify pages, you might want to consider FlexID or ProtoID so that users can set the page types, hence document structure, manually.

Ultimately, you must determine a page type for every single image entering your system, regardless of its source. Fortunately, the following are among the variety of techniques exist to determine the page type:

- Fingerprinting
- Recognition
- Order
- Content classification
- Manual

Not all identification methods are created equal, nor are they typically interchangeable. How you identify a page might limit how you can extract data from a page. For instance, identifying a page by fingerprinting ties that image to predefined zones, available for that page, so that you can use zonal recognition to obtain the data. If you use another method, such as recognition, the predefined zones are not available unless you create the trailing rulesets needed to read zonal information correctly.

Fingerprinting

Fingerprinting is most often used with form-based applications. With forms, the specific areas of the images that are to be recognized are known at the start of the project. When using fingerprint matching, the process gives you the three things that are required to use zonal recognition later on:

Page Type When a type is applied to a page, it determines what fields are created on that page to locate, store, and edit data.

TemplateID This is a number generated at run time and stored as a variable in the runtime DCO on every page that is matched by the fingerprint. This number ties the fingerprint to any zones that are defined for the fields on the page.

Offset When different instances of the same form are scanned, it is unlikely that the images are exactly aligned on each scan. Sometimes they are shifted slightly vertically or horizontally. This value (the difference between the fingerprint text and the runtime image text) is also stored in a page-level variable, and determines which direction, and how far, to move the zones stored on the original fingerprint when applied to the runtime image. This ensures that the zones defined on the fingerprint image align exactly to a shifted runtime image.

To use fingerprinting, the application must know the layout of the image that you want to fingerprint in advance, and add them to the fingerprint library. You also need to make sure that any image processing done to your runtime images is also done to the fingerprint images as they are added.

The application templates included in Datacap run the ImageEnhancement compiled ruleset automatically when adding new fingerprints. Make sure ImageEnhancement is configured according to your requirements before adding your own fingerprint images. If you must make changes to this rules set later, you might need to delete your old fingerprints and add them again.

If you have a lot of fingerprints to add, consider temporarily changing your `FindFingerprint(FALSE)` to a `FindFingperint(TRUE)`. This adds fingerprints for all new images that do not currently match one. In some cases, two or three fingerprints represent the same form page, but this is fine. Some forms have differences in the areas tested for a fingerprint match (this is configurable with `SetFingerprintSearchArea`).

There is no harm in having multiple fingerprints for the same form type. Fingerprints are only created if existing matching fingerprints are not found. In any case, you can edit the `SetFingerprint SearchArea` and `SetProblemValue` and run many different images to arrive at a fingerprint library that you prefer to use.

Recognition

The most common and reliable way of setting a page type based on recognition is using bar code recognition. Bar codes can quickly be found anywhere on the page, and seeing a page with a known bar code is an excellent way of identifying that page.

A less common approach is to use full or partial page recognition on each page and look for a unique phrase or combination of phrases. For example, finding “Form version 1.72” and “Page 1 of” together might allow you to identify that page. Multiple combinations can be tried for each page type, but it is important to test each new combination to make sure that a newly added identification method does not affect pages that were previously being identified properly.

If full page recognition on each image is necessary, there is no harm in doing it early (during the page identification phase) rather than later, after specific pages are identified. Although the ID of the page will be different, the data from the words on the page will be stored in the CCO file, which we can retrieve at any time.

Order (position)

Using order to determine page type means that you identify the image by its placement in the image stream relative to the position of other identified images. Although order can be used as a page identification technique by itself, it is often used in conjunction with other techniques.

For instance, if you see a bar code on a page you might want to set its page type to “Separator Page” and the following image in the stream to “Page 1.” Any page that follows a Page 1 is called “Page 2,” and so on. The Identify Pages ruleset has mappings that you can use to set this up.

Figure 5-6 shows the Page Identification Ruleset UI configured to identify images following Page to be named *Check*, and those following the Check images to also be named Check, until another Page image is found.

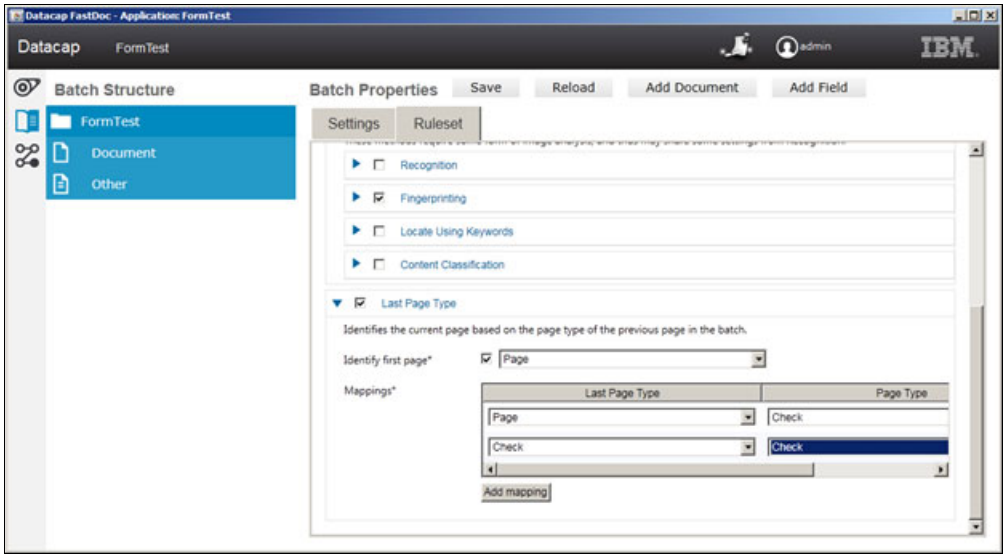


Figure 5-6 The Page Identification Ruleset UI demonstrating the sequential naming of pages

Content Classification module

IBM offers a Content Classification module to help identify pages and documents according to the content (words and phrases) in each document. Because each page is fully recognized, this is a slower process than many of the others, but useful when processing unknown documents with no other way to separate and identify them.

Manual method

IBM has several different Datacap clients, both thick and thin, that you can use to manually identify pages in a document. They are often configured to manually identify critical pages in the batch only, such as the first page of each document, and use the order technique to name the rest. This is typically done in a separate task immediately after a virtual scan but before any page identification or other processing takes place.

Figure 5-7 shows the FlexID user interface where the combination boxes can be used to manually identify pages.

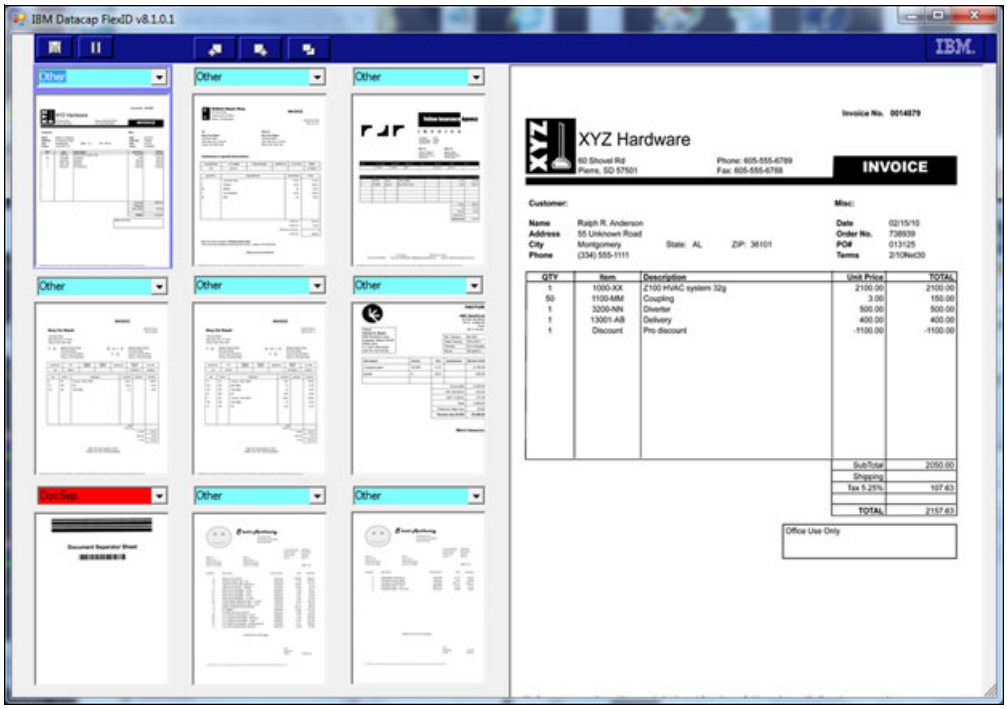


Figure 5-7 The FlexID display panel

5.3.3 Handling exceptions

Whatever methods you choose to do page identification, take the time to run many different images through each input path to make sure they work to your requirements. It cannot be stressed enough that a good page identification process is critical to the success of a project. Even though programmatic identification can be highly reliable, you still need to provide an exception path for images that are not properly identified programmatically.

In the Forms template starting application, exception processing is done through a Fixup process that operates when the PageID task ends with any unidentified pages or if the document structure is not complete. In most Forms applications, different pages from the document are treated as individual entities, each with a specific name and fields on the page, so it is important that we identify each individual page of the document correctly. If a page is misidentified or not identified at all, it needs to be corrected in Fixup to proceed.

Note: When using bar codes on form images, you normally must still perform a fingerprint match. This limits fingerprint matching to a specific page type (rather than the entire library of fingerprints), and can use a much lower match confidence value (`ProblemValue`).

In the Learning template, the data for the entire document is contained on the first page (Main_Page), regardless of which page the data is actually extracted from. As such, the pages do not need to be uniquely identified to process the document to extract data from it. The first page of each document gets a page type of *Main_Page*, and any following pages get a page type of *Trailing_Page* or *Attachment*. Trailing_Pages are pages that are recognized and can be searched for data to be stored on the Main_Page. Attachments are unrecognized images that can be exported with the rest of the document into an image repository.

Pages in a learning environment cannot generally be checked for document integrity until an operator sees the document at Verify time and verifies that appropriate document separation has taken place, and, if necessary, fixes the document structure in the Verify panel. If exceptions are encountered this late in the process, the problem documents are typically handled outside of the application, by automatically emailing the document to an appropriate person to take further action or by initiating some sort of business process management (BPM) workflow after the document is loaded into the image repository.

5.3.4 Extracting data from the images

There are four ways of extracting data from an image. They are ranked in order of preferable practice:

- Zonal
- Locate with regular expressions
- Locating by keyword
- Click N Key

Where possible, try to choose the extraction method that ranks highest in the list.

Zonal method

Zonal data is considered the best extraction method. You know exactly where the data you are trying to extract is located, and, knowing that position, you can be specific about how that area of the form is recognized.

In Figure 5-8, zonal techniques can be used to extract the data in a specific way. For example, the TOTAL field can be set to recognize hand printed characters; consider only numeric digits; and look for nine character positions. This gives you much better results than simply recognizing the zone without the additional information provided.

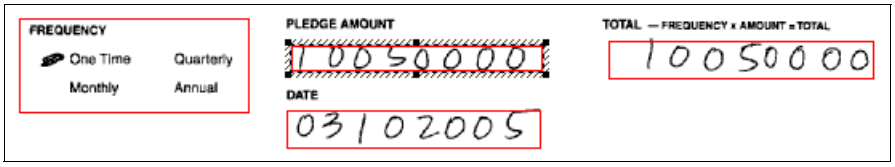


Figure 5-8 Zonal recognition example

Zonal recognition is not dependent on the format of the data (as is the case with Regular Expressions) nor on the successful recognition of data elements close to the desired data (as is the case with Keywords) to extract the data. You define a zone, and whatever is in that zone gets populated in the field.

Locate with regular expressions

This technique can find data that is located anywhere on the recognized page. When recognition takes place, a CCO file is created that contains the recognized data and the location where the data was found. If a zone is not known (eliminating zonal extraction), regular expressions are the next best choice.

A regular expression locates data by searching the entire CCO for data in the specified format. For instance, to find a date, you might write a regular expression to look for white space, followed by two digits, a dash, two more digits, another dash, and four more digits, see the following example:

```
RegExFind("[\b\s\n\r\^][0-9]{2\^][0-9]{2}[-][0-9]{4}")
```

Although regular expression syntax might look daunting, it is worth the trouble to learn the basics and to use this technique where applicable.

However, its use is limited to finding data that is uniquely and somewhat predictably formatted. Insurance IDs, banking account numbers, credit card numbers, phone numbers, and so on have a specific format that they must follow, which is unlikely to resemble other unrelated data in the document. However, if you are searching for a five-digit employee number on a document, it might be successful use of regular expressions for locating and extracting data can only be used in certain situations.

Locating by keyword

Keywords are labels that accompany the data you want to find on the form. For instance, you might have the data as shown in Figure 5-9.

Daily Rate:	$65.56
Optional Insurance:	$104.95
Taxes and Fees:	$150.02
Total:	$582.77

Figure 5-9 Sample keyword data

To find the various pieces of data in this example, it is possible to find text, such as *Insurance:,* and then look to the right or below that word to find the actual data that you would like to extract ($104.95).

Locating by keyword ranks third behind the zonal and regular expression methods in terms of reliability because you have to know what the keyword is and recognize the keyword correctly. You also have to anticipate the location of the actual data, relative to the position of the keyword, for extraction.

Keyword extraction is made easier by using keyword lists, which are text files that contain many different keywords that might be used as labels around the data you want to extract. For instance, a keyword list could contain `Total`, `Total Due`, `Pay this amount`, and similar words or phrases that occur in close proximity to the actual data.

The rules engine can be configured to account for the distance and direction that your data might be located relative to the keyword. For example, you could create a function that first looks to the right one word, and if the appropriate data is not found there, looks one line below the keyword.

Click N Key

The fourth and final method of extracting data from a form is for the data entry operator to click the form. Data at that location is then extracted and placed into the active field. This is the least preferred approach because it is the most expensive (it requires a human to do the clicking), although it is reliable.

An application is not limited to one method of extracting data. For forms, zonal recognition is generally sufficient because the zones are configured for each image type before use. However, in a learning environment, we typically use a hierarchy of techniques to find the data we are looking for.

It is common in a learning application for each field to try to find data in a predefined zone, and, if not found there, to try a regular expression or keyword locate. If those two methods also fail, the operator is prompted to click the data on the image. When the operator clicks the image, the data is extracted, but the position (zone) information of where the operator clicked is also stored and can be saved and used for future encounters with similar images.

This is why such applications are called “learning applications.” They automatically use the best extraction technique available on every image, and they can use the process to learn how to best extract data from future images. Rather than knowing what every image looks like at the outset of the project, such applications add fingerprints during runtime and use zonal recognition on an increasing percentage of the images encountered over time.

5.3.5 Getting data that is not on the form

Many projects require data that is not actually on the form. In some cases, data from a more reliable database lookup is chosen over data obtained from the form. In other cases that data might result from a calculation involving form data even though the form data is not required.

Database lookups can be done before data entry, using a background process, or during data entry, with or without a user interface. The operator is able to pick a single record when multiple records are returned.

Figure 5-10 shows the DCDesktop Verify panel presenting the database lookup data to an operator.

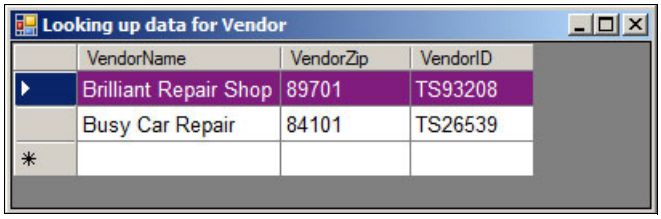


Figure 5-10 The DCDesktop lookup panel

Where data should return a single record, such as a Name associated with an employee number, you can put a button or an event on most data entry panels to do the lookup using a ruleset and not display a UI. This is similar to running a lookup in a background process, but the UI has the event or button to call that ruleset and fill the data without forcing the operator to select from multiple results.

The ability to use validated sources of data to aid data in extraction or validation is often overlooked. For instance, using POLR in the APT application, line items on purchase order (PO) invoices are automatically matched based on the PO record in the enterprise resource planning (ERP) system on price, quantity, and item number. These values are mostly numeric and recognize well, whereas item descriptions often contain punctuation and can recognize with errors or low confidence characters. However, if there is a POLR match, the item description can be updated using the description in the ERP system.

5.3.6 Validating the data

Data validation is extremely important. Without adequate data validation, bad indexes or data can be saved in the target repository. Export processes might even fail because of a bad data type.

In general, every field on the form should go through a validation process. Every data and image repository has restrictions on data it can accept for fields that you want to export to. Maximum lengths and data types are common restrictions, but there might be others. In the validation ruleset, you must make sure that you check each field to ensure it can be exported properly and contain the correct data.

Validations are designed to be checked before a data entry operator sees the data. This way, when a page or document is viewed by a data entry operator, all data that has failed validation is flagged for them to review. The same validations are normally run after the data entry operator submits the page. This guarantees that the export to the repository is accurate and will not fail because of an improper length or data type.

Data in a particular field might have several valid formats. A United States postal code, for example, might be either five digits or five digits followed a hyphen (dash) and then another four digits (12345-1234). If you are validating US postal codes, allow for both conditions.

Two other factors must be considered for each field:

- Whether you will allow a data entry operator to successfully submit the form if the conditions you specify are not met
- Whether the field can be blank

Figure 5-11 shows a field that cannot be overridden. It can be five digits, nine digits, with a dash in the sixth position, or blank.

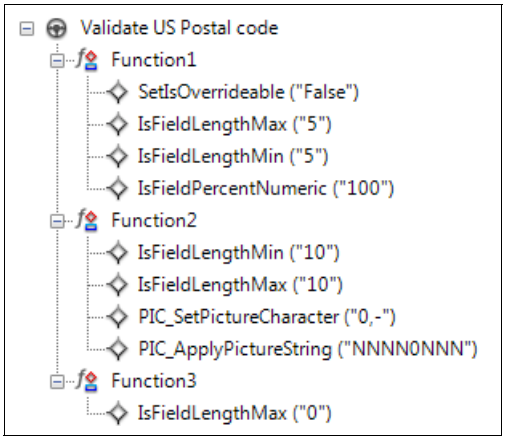


Figure 5-11 Field validation example

It is critical that you go to this level of detail when validating data to ensure good quality data for your application. Currently the compiled validation ruleset does not support multiple conditions and should be used only when your validation needs are basic. Do not fall into the trap of making complex validations more general to satisfy multiple conditions. For instance, for US postal codes, you could specify a single condition that says the value is 0 to 10 characters and at least 90% numeric. All valid US postal codes will pass this rule, but so will a lot of other values that are not in a valid format.

5.3.7 Verifying the data

Currently, three “client” programs can present data and images to a data entry operator for verification. In this book, we cover the thick client, Datacap Desktop, and two thin clients, Datacap Web Services and IBM Content Navigator. Over time, the capabilities of each might be enhanced and each program has different capabilities. For basic projects, these clients can be used interchangeably, but more complex projects might limit you to one or two of the offerings.

The choice of which client to use is often determined by customer preference, before the design phase of the application. This can limits what you can do, and how you must do it. When requirements and client preferences clash, something has to change. You should be aware of all requirements and check them against any restrictions in the chosen Verify method before you start the project.

One crucial data entry requirement that is often overlooked until the project goes live is that most data entry operators much prefer using keyboard shortcuts to verify and edit data. For example, many first-time application designers put graphical user interface controls, such as nice buttons, on the verification forms, thinking that it makes the data entry process easier. It usually does not. Instead, users often request that such buttons be removed in favor of a keyboard shortcut to activate the feature. Depending on the product that used for the verification process, you must learn and implement the available keyboard shortcuts to activate events as much as possible.

5.3.8 Exporting the data

Now we have come full circle. If you followed the suggestion of collecting your export requirements at the beginning of the process, the field selection and validations you have defined in the application should support exporting to whatever system you choose. Datacap has several compiled rulesets and many different action libraries to carry out the actual export process.

The final step in your application is to handle documents that fail export. Even though you have made sure the repository should not reject any data that gets to this point in the process, the repository might be unavailable because of maintenance or a network outage. In such a situation, your image and data cannot make it to their final destination.

Typically, the export libraries and rulesets will abort the batch when this happens, allowing an automated process (such as Datacap Maintenance Manager) to retry the export at a later time. In some cases, such as in applications built using the Learning template, data entry operators might mark documents for Rescan, Review, or Deletion, and those exceptions must be handled on a document by document basis. Sometimes, the requirements dictate that such documents go to a BPM workflow, but sometimes they can be emailed to someone, instead. It is also possible to add routing to the export process so the batch is routed to a supervisor for deletion, approval, or to make changes in a repository (such as adding a vendor record). When such steps are complete, the export can be retried. In the case of a Rescan, the batch can be routed to a Fixup operator who might replace one or more images and be routed back to the Verify process.

Always make sure that you know the exception requirements before you start developing your application. Many of your processing choices that you make are dependent on how you deal with exceptions.

5.4 Configure external processes

Two external processes that are normally set up for every Datacap application are Datacap Report Viewer and Datacap Maintenance Manager.

Datacap Report Viewer is used by supervisors to view system and operator performance over a period of time.

Datacap Maintenance Manager is a scheduled process that is used to delete batches automatically, send notifications, and even reset batches that are aborted because of network outages or for other reasons.

For more information about configuring external processes, see the Datacap section in the IBM Knowledge Center:

- Datacap Report Viewer

  http://ibm.co/1LOzzBe

- Creating a Datacap Maintenance Manager application

  http://ibm.co/1MlnERJ

6

Structured forms application

Datacap is a versatile capture solution that is able to handle both structured and unstructured documents. A *structured* document is a document, such as a form, where every instance contains the same type of data in the same location on the page, such as a loan applicant's name or social security number. An *unstructured* document might have data in different locations depending on the length of the document or the amount of text in a particular section. An example might be the date a contract is signed or a bank statement, where the account balance appears at different distances from the top of the page, depending on the number of transactions listed.

Datacap can also process *known* documents, which are documents created and published by the same organization that is capturing the data, such as an insurance claim form or tax form. Similarly, Datacap can process documents that it has not scanned before, such as an invoice from a vendor, when it is known *what* data must be captured. However, because every vendor has a different invoice format, it is not known *where* that data is located on the page.

Datacap supports several techniques so it can "learn" (and remember) how to capture data from unknown documents when it scans such a document for the first time. See Chapter 7, "Unstructured document application" on page 147 for a description about building a learning application using the *Learning* template.

In this chapter, we describe how to create a Datacap application that is optimized to extract data from known, structured documents, such as forms created by the same organization capturing the data. We use the *Form* template, which ntroduced in Datacap 9, to quickly build our application.

This chapter covers the following topics:

- Scenario background
- Configuring the Datacap application
- Testing your new forms application

6.1 Scenario background

A fictitious lending company is running a marketing campaign try to grow its mortgage business by getting mortgage holders to refinance at a lower rate. It has printed and distributed a postcard asking for basic contact information and details about the potential customer's current loan. The postcard carries a bar code, which is used to identify the specific form type and can also serve to track which newspapers or periodicals are used to distribute the form, providing metrics about the success of the campaign. The prospective customer enters data by hand into *constrained* fields. Datacap uses intelligent character recognition (ICR), optical mark recognition (OMR), and bar code recognition to extract the data.

Figure 6-1 shows an example of the completed postcard.

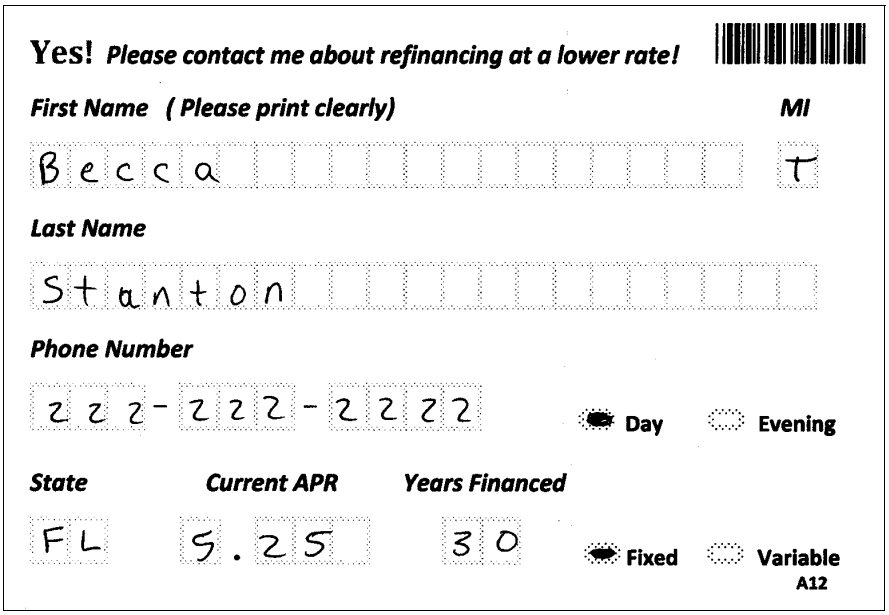


Figure 6-1 Sample document

Although in this example we process a paper-based form that has been filled out by hand, the techniques used in the form-based application that we describe in this chapter apply equally to forms with machine print and forms that are created and processed electronically.

6.2 Configuring the Datacap application

In earlier versions of Datacap, application developers often copied an existing application to use as the starting point for a new one. Datacap 9 introduces the concept of an *application template*. At the time of writing this book, two templates are included to start application development:

- FormTemplate
- LearningTemplate

In this chapter, we build an application by using FormTemplate.

6.2.1 Creating a new structured form application

We use the Application Wizard to create a new application. The Application Wizard can be accessed from Datacap Studio and FastDoc (Admin). In this chapter, we primarily use FastDoc (Admin) for our examples.

> **Note:** For an overview of Datacap administration clients, see 3.2.2, “Administration clients” on page 70.

Use the following steps to create a new structured form application:

1. Open **FastDoc (Admin)** in Local mode and click the Application Wizard icon at the upper-right.

   > **Note:** Although you could simply open the FormTemplate application directly, it is advisable to create a new application using the Application Wizard instead. This method keeps the template applications unmodified and available for future use.

2. Click **Next** on the Overview page and select **Create a new RRS application**, as shown in Figure 6-2.

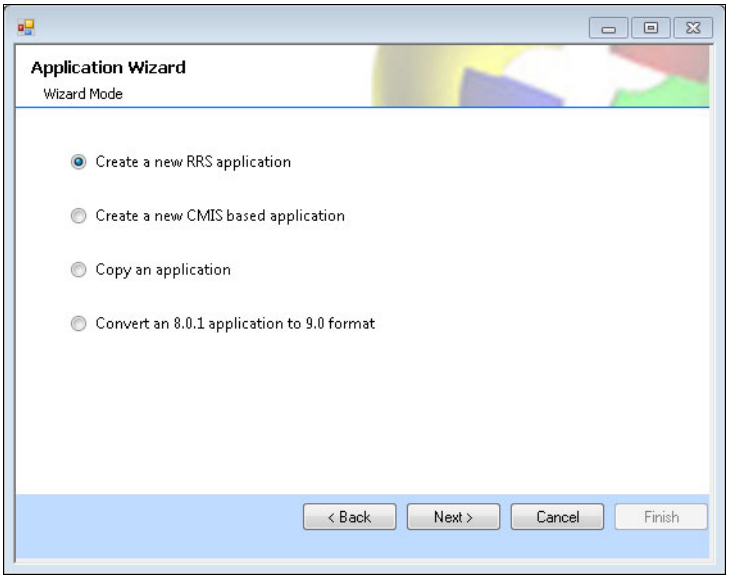


Figure 6-2 Create a new RRS application

It is also possible to create a new application using Datacap 9’s Content Management Interoperability Services (CMIS) support. If you have a CMIS-enabled repository, you can select this option to quickly load page and field definitions into your application. In this chapter, we assume that you are created an application from scratch, without using CMIS.

3. Give your new application a name, **Datacap Form**, and select **FormTemplate** from the Application template drop-down menu, as shown in Figure 6-3.

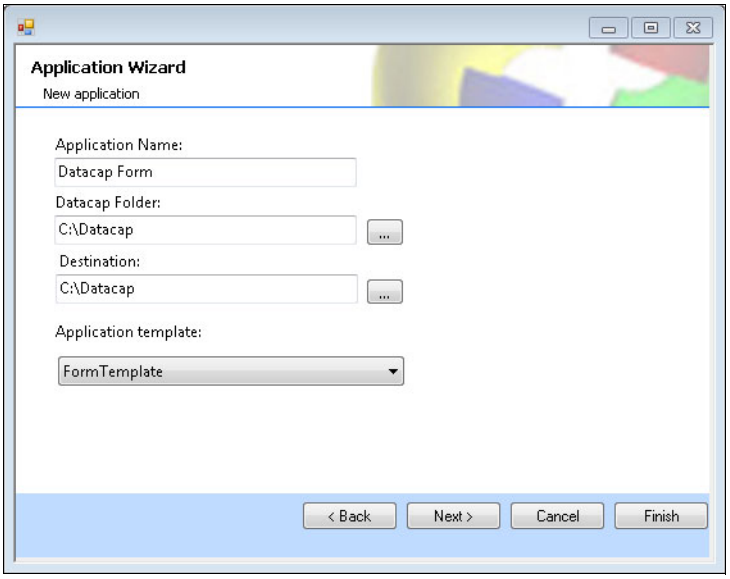


Figure 6-3 Enter a name of the new application

4. Click **Finish** to accept the remaining default options and to complete the application setup process.

Note: You can also convert one of your applications into a template. Simply add it to the Datacap Templates folder, for example `C:\Datacap\Templates`, to take advantage of this capability.

5. Close **FastDoc (Admin)**, reopen it, and log in to your new, server-side application, using the default user name and password `admin/admin`, as shown in Figure 6-4.

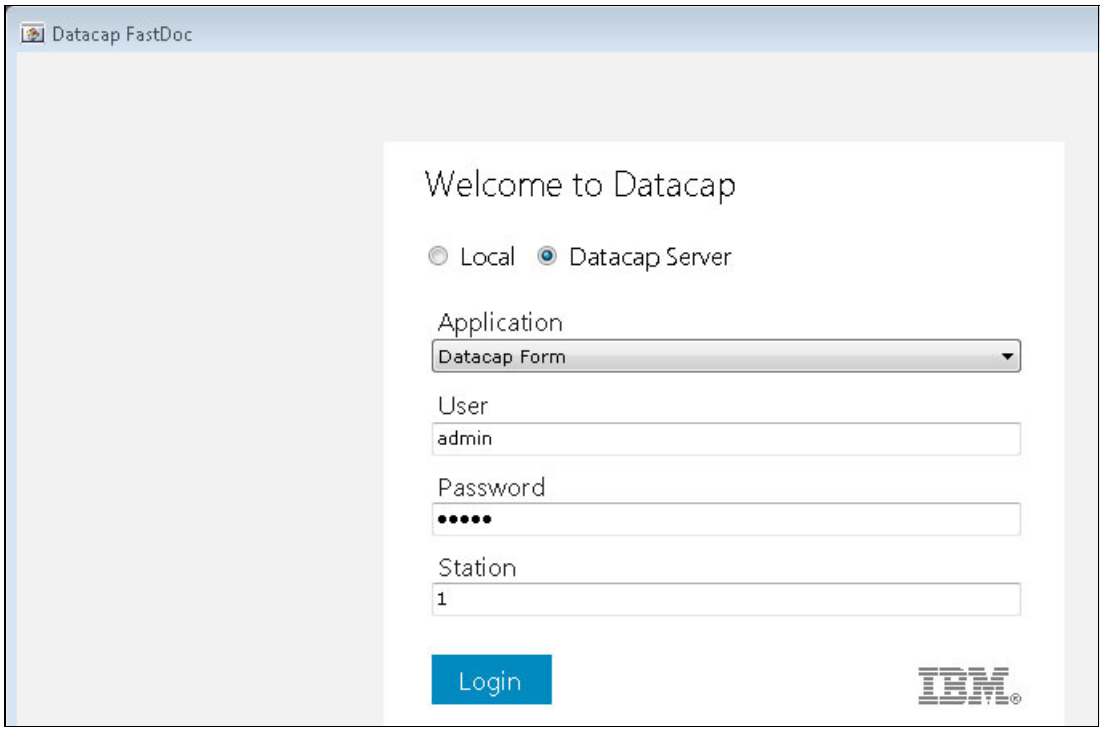


Figure 6-4 Log in to your new application

6.2.2 Workflow jobs

The new application Datacap Form, created using the FormTemplate template, contains six workflow jobs. These workflows are accessible by clicking the **Configure Workflow** icon on the left, as shown in Figure 6-5.

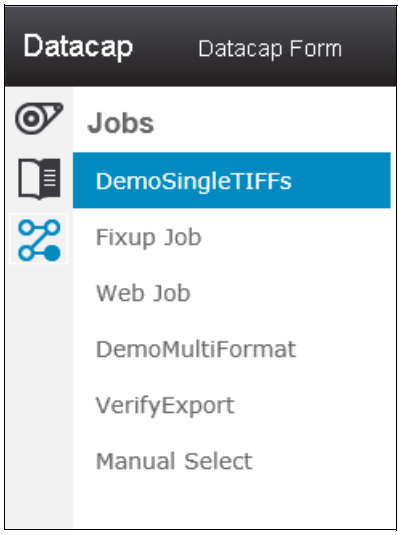


Figure 6-5 Workflow jobs in Datacap Form application

DemoSingleTIFFs

This workflow reads single TIFF image files from a directory on disk, `C:\Datacap\Datacap Form\images\Input_SingleTIFFs`, and processes them using the following workflow steps, each containing one or more rulesets to do the work:

Vscan	Reads files from a disk
PageID	Cleans images, identifies pages, and creates documents
Profiler	Recognizes data and runs validation rules
Export	Exports data and images

In Figure 6-6, the workflow steps used by this job are shown in dark green. The light-green boxes denote the rulesets within each step. Rulesets with an ellipsis (...) can be edited by double-clicking them. Those that cannot be edited in FastDoc (Admin) can be modified in Datacap Studio.

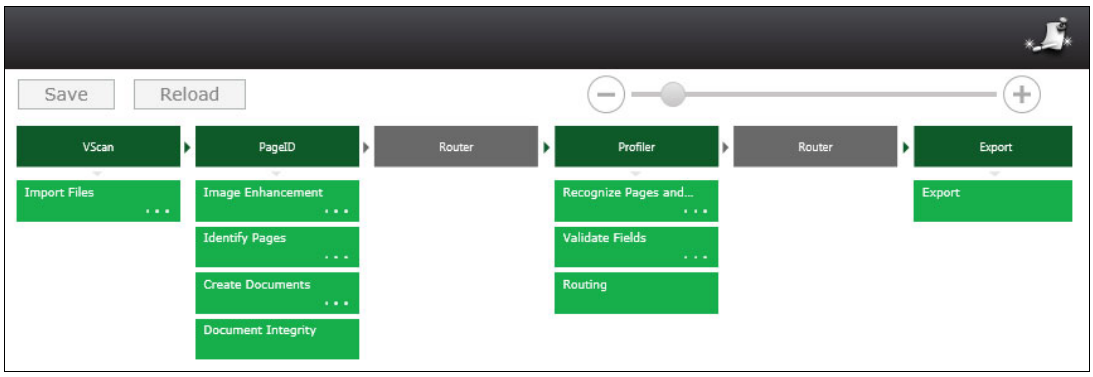


Figure 6-6 Five-step workflow in DemoSingleTIFFs

Two optional jobs might be started, depending on the results of the *PageID* and *Profiler* tasks. This is indicated by the *Router* step, as shown in Figure 6-7.

Figure 6-7 Router step

In the PageID step, the Document Integrity ruleset splits off any documents where the document integrity is incorrect or where there are still pages of type Other. These are sent to the Fixup Job, where an operator must identify the pages manually.

Similarly, in the Profiler step, the Routing ruleset splits off and high-confidence documents and sends them straight to export, while sending low-confidence documents, which need an operator to verify or validate the data, to the VerifyExport job. For example, in a 50-page batch, if only 5 pages need verification, this ruleset sends the majority, 45 pages, to export, which allows scanning and data capture to continue uninterrupted.

DemoMultiFormat

The DemoMultiFormat workflow is almost identical to DemoSingleTIFFs. However, it is preconfigured to support files that are not in TIFF format, such as Microsoft Word documents or JPG image files. To do this, the first step is to edit the compiled ruleset called Convert Files to Images in the VscanMulti workflow. You can edit the ruleset by double-clicking it. Aside from the initial document ingestion step, the remainder of the workflow is identical to DemoSingleTIFFs.

Note: Compiled rulesets are rulesets that offer a pre-set list of actions that can be easily configured through a single interface. In FastDoc (Admin), compiled rulesets display an ellipsis (...) in the ruleset icon.

Web Job

This job is configured for use by the web client, Datacap Web Services. Aside from the initial document ingestion and upload steps, the remainder of the workflow is identical to DemoSingleTIFFs.

Note: An application built using the FormTemplate template can also be configured to run in IBM Content Navigator.

Manual Select

This job supports using a physical scanner attached to the computer rather than reading the image files in electronic format the way the other three jobs, DemoSingleTIFFs, DemoMultiFormat, and Web Job do.

Aside from the initial document ingestion step, the remainder of the workflow is identical to DemoSingleTIFFs.

6.2.3 Setting up the document, pages, and fields

Follow these steps to set up the documents, pages, and fields that your application must support:

1. Click the **Configure documents, pages, and fields** icon at the left. Figure 6-8 shows the user interface for configuring the document, pages, and fields in our application.

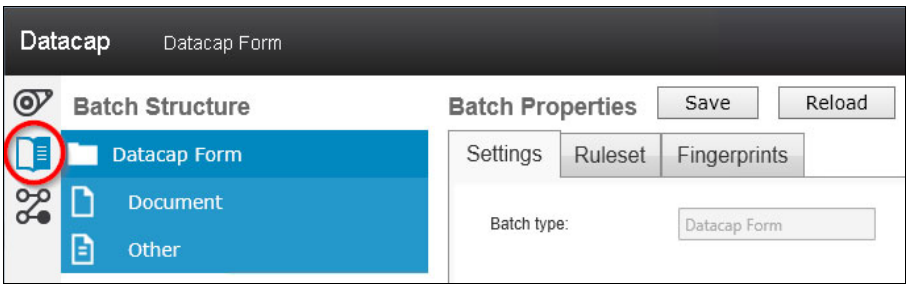


Figure 6-8 Configure document, pages, and fields

2. With the application name selected, click **Add Document**. This brings up the Add Document dialog window shown in Figure 6-9. Give the document a name, such as `Postcard`. Check **Enable** and select **Document** from the drop-down menu to enable inheritance of default rulesets from the “Document” (default) document. Click **Add** to complete the process.

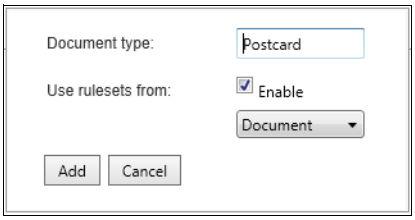


Figure 6-9 Add document

> **Note:** Do not use or modify the default Document and Page objects in any way (for example, by adding pages or fields). These objects contain application-wide default rulesets that can be used, through inheritance, in documents and pages that you create. This saves you from having to add the default document and page-level rules to every page that you add to your application.

3. Select the added document and click **Add Page**. As before, give the new object a name, `CardBack` for example, and enable inheritance by configuring the page as shown in Figure 6-10.

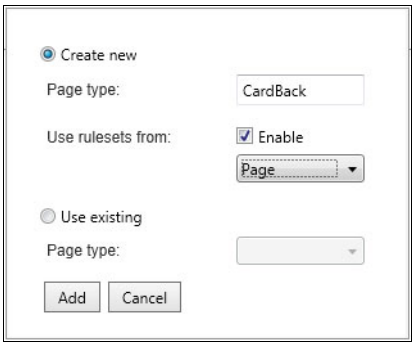


Figure 6-10 Add page

4. Add fields to contain the data we want to capture from our postcard. The fields correlate with the data on the postcard shown in Figure 6-1 on page 132. Specifically, the following fields must be added to the page:
   - Campaign
   - FirstName
   - MiddleInitial
   - LastName
   - PhoneNum
   - CallTime
   - State
   - APR
   - YearsFinanced
   - MortgageType

Adding fields is much like adding pages and documents. With the page selected, click **Add Field** and enter the name of the field. In our example, we do not use inheritance for the fields on our page.

5. Enable OMR.

   Most of the fields on our page require no additional configuration, therefore, we skip field-level validation for now. However, two of the fields do require additional configuration because we use OMR to capture the data: CallTime and MortgageType.

   To enable a field for OMR for CallTime, select the field to display the Settings tab. **Enable** Optical mark, and then click **Add** to add two value and display combinations, as shown in Figure 6-11.

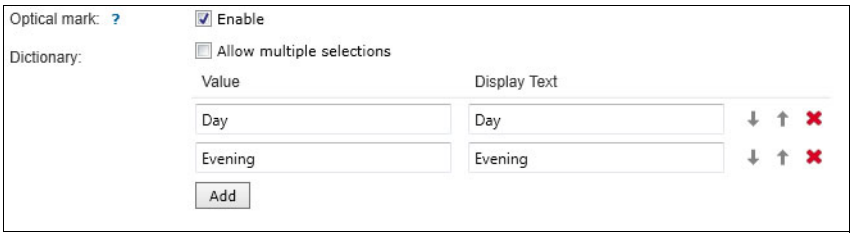


Figure 6-11 Enable OMR

   Do the same for MortgageType by using the follow value and display combinations:

   – Fixed/Fixed
   – Variable/Variable

6. There is a third piece of data on the document that we need to use, the bar code, but we configure bar code recognition later.

7. The last setting that must be configured page level is the maximum and minimum number of times a page is allowed to occur in the document and its position. Click the **CardBack** page and, on the Settings tab, set all values to **1**, as shown in Figure 6-12.

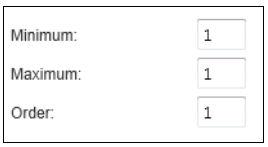


Figure 6-12 Page minimum, maximum, and order

6.2.4 Configuring rulesets

The great benefit of using an application template is that much of the necessary setup and configuration to get to the prototype stage is done for you. As such, in additional to adding fingerprints as described in 6.2.5, “Adding fingerprints” on page 141, only two rulesets need to be edited to use the application:

- “Image Enhancement” ruleset
- “Recognize Pages and Fields rule set” ruleset

You can configure a third ruleset, Validate Fields, if you want to enforce business and formatting rules to your data. For our purposes, we do not use field-level validation.

Image Enhancement

Unless a document is in pristine form, which is rare for scanned paper images, the document needs to be cleaned and enhanced to improve the accuracy and reliability of the recognition (OCR) process. For example, documents might be skewed or contain speckles that can affect how reliably the recognition engine “sees” text, handprints, and other markings on the page.

FastDoc (Admin) provides an interactive user interface to test out image cleaning and enhancing. This is useful because you can quickly configure the most appropriate cleanup and enhancement actions for the specific documents that you need to process in your application.

In “Configure documents, pages and fields,” click the **CardBack** page and then the **Ruleset** tab. Select **Image Enhancement** from the drop-down list. This displays the image enhancement options, with a check mark next to those that are enabled and configured. Additionally, two panes are displayed on the right. The one on the left contains the image that you are working with, with no modifications, and the one on the right immediately reflects any changes you apply to your image. Load one of your images by clicking **Open image file** under Image Operations. Notice how the original is displayed in the right side pane while the enhanced image is displayed on the left, as shown in Figure 6-13.

Note: Rulesets are context-dependent. That is, they operate on objects in the document (DCO) hierarchy: some operate at document level, others at page level, and so on. When you configure a ruleset, you must have the appropriate DCO object selected. If you do not, FastDoc (Admin) displays a warning to this effect.

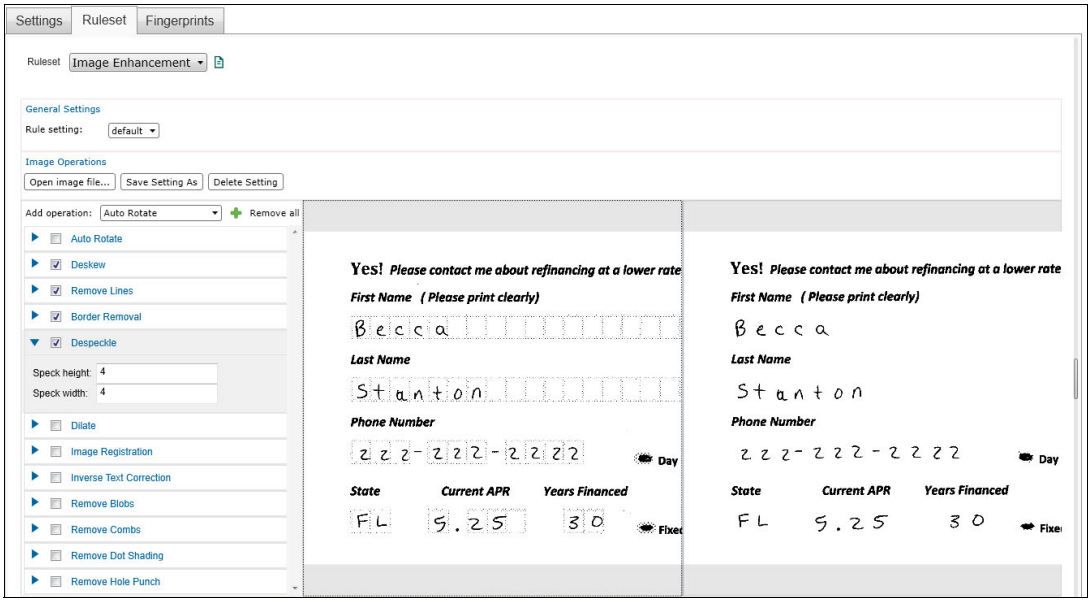


Figure 6-13 Image enhancement configuration

By default, the most commonly needed enhancement options are enabled and configured. However, because every document is different, these options might need to be adjusted or additional options might need to be configured and enabled. In our case, the postcard contains constrained fields, denoted by dotted boxes, so we want to make sure those are removed, leaving only the hand-printed text.

To do this, expand the **Despeckle** option and set the values to **3**. Also expand the **Remove Line** option and change the minimum length to **100** to ensure that the bar code is not removed.

> **Note:** When configuring image enhancement, it is important not to “lose” important information about the page by configuring the options to be too aggressive.

Recognize Pages and Fields rule set

The second ruleset that you must configure is Recognize Pages and Fields. Unlike Image Enhancement, this ruleset applies at both page and field levels. However, because we do not do full-page OCR so we only complete the configuration for each of the fields on the page.

To enable the Campaign field, click the **Campaign** field under CardBack and check **Read Field**. Because this is a bar code, we enable bar code recognition, as shown in Figure 6-14.

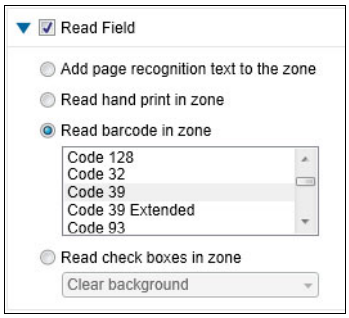


Figure 6-14 Read bar code field

Enable field-level recognition for the remaining fields on the page as well. Because those fields contain hand print, select **Read hand print in zone**.

For the two OMR fields, CallTime and MortgageType, select **Read check boxes in zone with Clear Background**.

6.2.5 Adding fingerprints

After adding documents, pages and fields to the application, the next step is to add a fingerprint. Applications built using the FormTemplate use Datacap’s quick and reliable fingerprinting technology to identify pages. Zonal recognition can then be used to identify areas on the page for data extraction.

> **Note:** Although it is possible to use another mechanism for page identification, such as bar code identification, doing so would require additional changes to the application to make sure the hand printed fields can be read zonally. Hand print can only be read zonally.

To add a fingerprint, use the following steps:

1. In FastDoc (Admin), click the **Configure documents, pages and fields** icon on the left. Click the **Fingerprints** tab and, under Fingerprint Class, click **Add**, as shown in Figure 6-15, to add a new fingerprint class called `Postcard`.

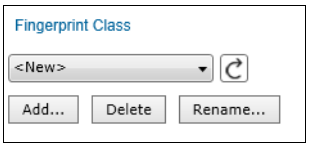


Figure 6-15 Creating a new fingerprint class

2. Next, click **Add** under Fingerprints and select your document from the file system. This displays your image on the right and creates a (numeric) fingerprint. Select **CardBack** from the drop-down menu under Page Type to associate this fingerprint with the CardBack page, as shown in Figure 6-16.

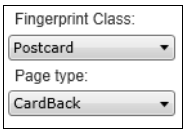


Figure 6-16 Associate the fingerprint with CardBack page

Notice that the image enhancement settings you configured earlier are applied to the fingerprint image also.

3. For each field on the page, with the exception of CallTime and MortgageType, add a zone to the fingerprint. First, select the field in the DCO, for example **Campaign**, and then draw a box around the corresponding data on the image (the bar code in this case) by holding down the left mouse button above and to the left of the bar code and releasing it below and to the right. Figure 6-17 shows the result.

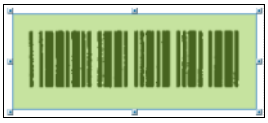


Figure 6-17 Creating a zone for the bar code

Repeat this process for each field on the page (except for the two OMR fields). Keep in mind that the field on the printed form might extend beyond the data on this particular instance of the completed form. Make sure that your zones are wide enough to accommodate all data allowed in a field on the paper document.

4. Enable OMR.

For the OMR fields we use Datacap Studio. At the time of writing, Datacap Studio provides several features for OMR fields that are not yet available in FastDoc (Admin). Close **FastDoc (Admin)**:

a. Open Datacap Studio and log in to the **Datacap Form** application.

b. On the Zones tab, open the PostCard fingerprint class and select the **CardBack** fingerprint.

c. Select the **CardBack** page in the DCO, as shown in Figure 6-18.

Figure 6-18 Select page in DCO on Zones tab

d. As you click each field the fingerprint image, on the right side, displays all zones on the image and highlights the one corresponding to the currently selected field. The two OMR fields have not yet been zoned. Select **CallTime** and draw a box around the entire area that contains the *Day* and *Evening* labels and the OMR boxes.

e. Next, with the **CallTime_OMR1 (Day)** sub-field selected, draw a box around the OMR field on the image, as show in Figure 6-19.

Figure 6-19 Create box for OMR field on the image

f. Do the same for the second OMR sub-field, CallTime_OMR1 (Evening). Make boxes the same size by Ctrl + clicking both boxes, and then right-clicking in the middle of one of the two boxes and selecting the option **Make same size > Both**.

> **Note:** It is important for OMR boxes (containing options or choices that logically belong together) to be identical in size. When the recognition engine evaluates each box, it looks at the pixel density to determine whether an option has been selected or not. Unevenly sizes boxes can have differing pixel densities because of their difference in size, which can lead to inaccurate results.

g. Repeat this process for the second OMR field, MortgageType. The fingerprint and zones are saved automatically.

6.3 Testing your new forms application

After you have completed the new application, you can test it using the “Process batches of documents” icon on the left side in FastDoc (Admin), as shown in Figure 6-20.

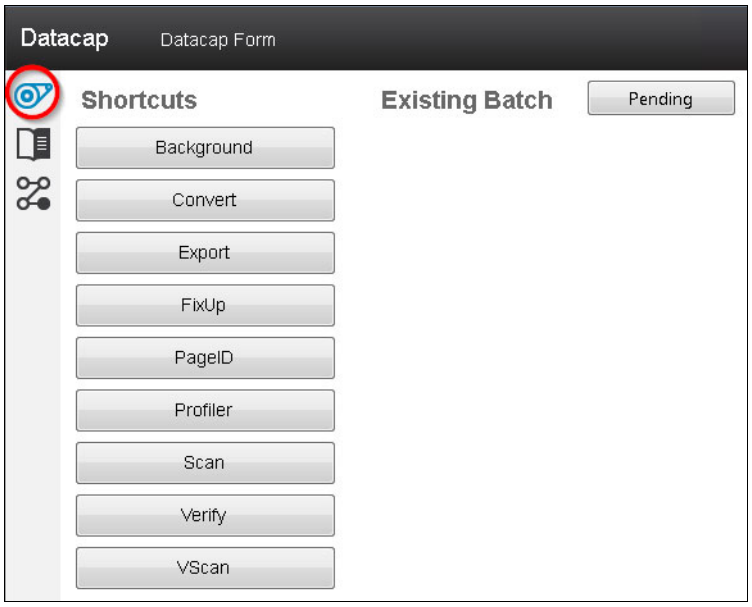


Figure 6-20 Process batches of documents

Note: You may also use other clients to test your application, such as Datacap Desktop or the web client Datacap Web Services. See 3.2.1, “User clients” on page 69 for an overview of Datacap clients.

To initiate the process, click **Vscan**. Next, you are prompted to select whether you want to process TIFF images or images in multiple formats, such as PDF or JPG, as shown in Figure 6-21. Documents are read from the designated input folder, `C:\Datacap\Datacap Form\images`.

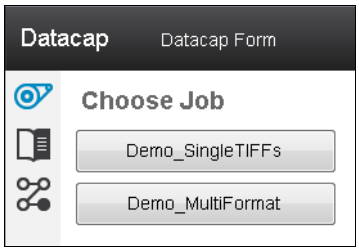


Figure 6-21 Select Demo_SingleTIFFs or Demo_MultiFormat

Because your new application has not been configured to run background workflow processes automatically, such as PageID, Profiler, and Export, you must initiate those processes manually, either by clicking **Background**, which runs the next background task profile in the workflow, or by clicking each shortcut:

- PageID
- Profiler

To check whether any of your test documents require verification, in case the OCR process returns a low-confidence indicator for one or more documents, click **Verify**. This opens the verification panel as shown in Figure 6-22.

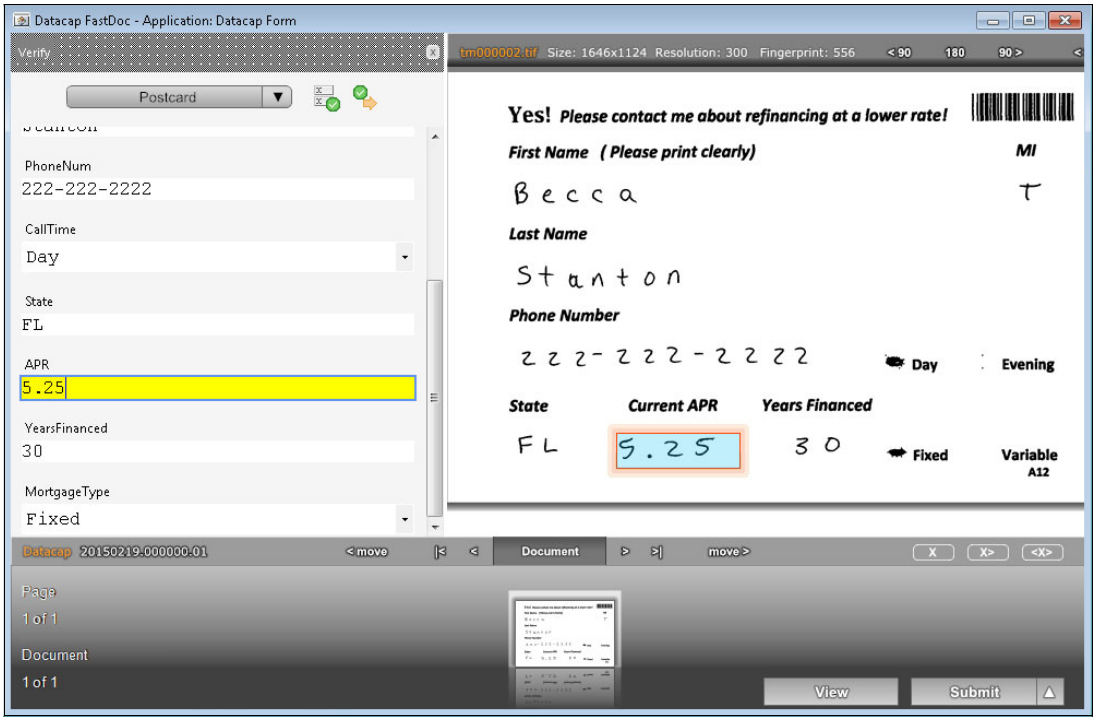


Figure 6-22 Verification panel

After any confidence issues were resolved by manual correction, click **Submit** to proceed to the next step in the workflow, which is Export in this case.

Also, either click **Background** a final time, or click **Export** to run the task yourself to complete batch processing. See the `*.log` files in the appropriate batch folder for additional information about how the documents, pages, and fields were processed through the workflow. For the `*.txt` files that contain the data extracted from your sample documents, see the `C:\Datacap\Datacap Form\export` export folder.

7

Unstructured document application

The Learning template is for use when you have unstructured or semi-structured documents in your application. It is also possible to use machine printed and bar-coded forms in addition to your learning component. In this chapter, we demonstrate both in the scenario.

This chapter covers the following topics:

- Scenario description
- Selecting the Learning template
- An overview of the Learning template
- Building the document structure for the MktBankStmt application
- Configuring the Learning template for the MktBankStmt document structure
- The routing ruleset
- Testing with a validation panel
- Exporting
- Wrapping up your project

7.1 Scenario description

After Bank A receives and processes their marketing campaign cards, they call the potential customers to explain more about their offerings. If the customer is interested, the Bank A representative creates a case to track the progress of the potential refinance. As part of the case, the potential customer is to supply a current bank statement that shows that they have the means to cover the refinancing costs plus any additional payment that the bank requires for refinancing.

Bank A sends a letter to the potential customer. It has a bar-coded reference to the customer number assigned to the case, with instructions to send the letter back with the current bank statement. Because the letters are all similar and change only with the bar code value and the address printed on the letter, and because the information is always in the same place, this document is considered a *form*.

However, the bank statements are a different matter. Bank A has no control over the format of these documents, and there could be thousands of different formats, yet all containing the required information that the bank wants to capture from them. For the bank statements, the *Learning template* is the right choice.

With the Learning template, each time a new format for any bank statement is received, it is analyzed to try to automatically find the data through locate rules. What cannot be found will be presented to an operator to click the appropriate place on the image where the data is found. By a combination of automatically finding data locations by rules and clicks by the operator, the system remembers where that data is found so that it can use *zonal* extraction method the next time that a similar statement from that particular bank is encountered.

For this example, we treat the letter as a separator sheet and capture the bar code from it. This bar code is used to look up the name and address of the customer so that it can be displayed on the data entry panel at the same time that the bank statement is displayed. By doing this, the operator can visually check that the bar code was read correctly and that the names and addresses on the bank statement match the Bank A records.

For the statement, our case management system requires the institution name, account number, date, and balance on the statement.

Now that we know what we need to export, we explain how you can begin designing a system to capture it.

7.2 Selecting the Learning template

Opening FastDoc in the administration mode enables you to use the Application Wizard to create a new learning application. The Application Wizard icon is on the menu bar. After selecting that you want to create a new RRS application, make sure that you name the application appropriately and choose **Learning Template**.

For this example, we use the name `MktBankStmt` for the application as shown in Figure 7-1.

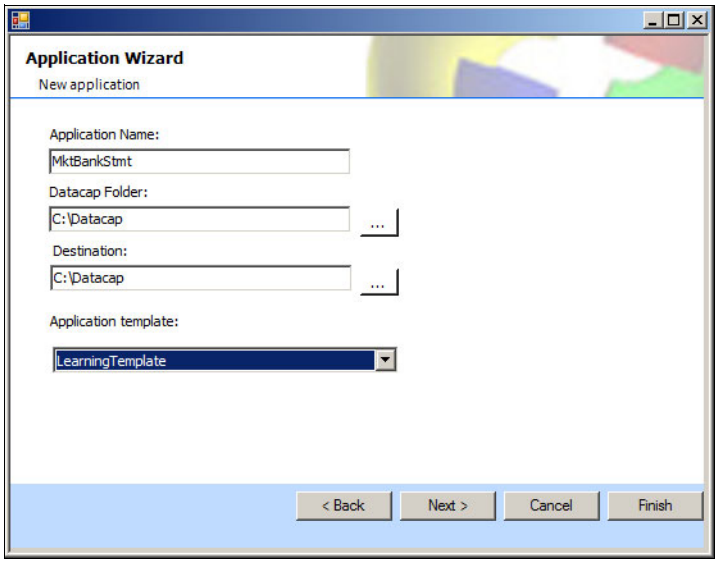


Figure 7-1 Using the Application Wizard to create a new RRS application with the Learning template

After naming and selecting the proper template, click **Finish**. The choices that follow can now be accomplished with FastDoc in a more visual and interactive mode.

Exit FastDoc and log in to your new application as user `admin`, `station 1`, with a password of `admin`.

7.3 An overview of the Learning template

On the left icon bar in FastDoc, choose the **Configure Workflow** option. You should see a window similar to Figure 7-2.

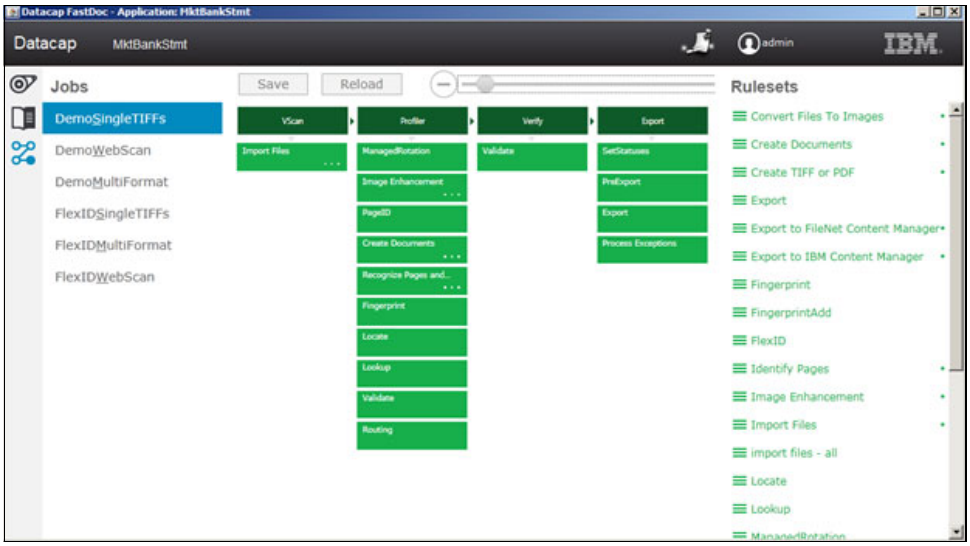


Figure 7-2 The Configure Workflow window of the example learning application

You do not make any changes to these workflows now, but you need to understand what each component does to master the use of the template.

7.3.1 The Learning template jobs

At the time of this writing, there are six jobs available for you. They are all similar in processing, yet they differ in the types of images that they can process and how the images are ingested into the system:

- DemoSingleTiffs

  This job takes single page TIFF images from the `\datacap\appname\images\Input_SingleTIFFs` folder. Even though it takes single TIFF images, by using the separator sheet from the APT project, you can input multiple page documents. Verify is set up to use DCDesktop, Datacap Web Services, and FastDoc clients.

- DemoWebScan

  All processing and verification is the same as DemoSingleTiffs, but the input is done with Datacap Web Services with an additional Upload task.

- DemoMultiFormat

  This job takes every image that can currently be converted from the `\datacap\appname\images\Input_MultFormat` directory. The processing difference is that it uses the input image to determine document structure, one document per image. For instance, if you put in a three-page PDF image, all three pages will be treated as one document.

- FlexIDSingleTiffs

  This is identical to DemoSingleTiffs except that a FlexID task is inserted between scanning and processing to manually identify pages.

- FlexID MultiFormat

  This one is identical to DemoMultiFormat but with the FlexID task inserted.

- FlexIDWebScan

  This is identical to DemoWebScan but with ProtoID inserted to manually identify pages before processing.

7.3.2 The Learning template tasks

The green boxes in the middle of Figure 7-2 on page 149 (shown previously) identify the rulesets that are in every task:

- Import Files

  This task contains the compiled user interface (UI) ruleset to input files from the disk.

- Convert Files to Images

  This one is not shown in Figure 7-2 on page 149, but it is available by clicking either of the MultiFormat jobs. These are ready for use and configured to convert all image types that Datacap can currently convert to 300 dpi Group 4 bi-tonal TIFF images for processing.

- ManagedRotation

  This task automatically rotates the images with the ScanSoft recognition engine. It is run in a managed fashion, so if there is an error, the recognition engine automatically restarts and continues processing.

- Image Enhancement

  This task enhances the image by deskewing, removing lines, shaded backgrounds, and so on. Be conservative when setting this up, becauses it will apply to all images. In this example, we must make sure that it does not erase any lines in the bar code.

- PageID

  This identifies pages according to the order of known pages. By default, it is expecting particular document separator sheets (those used with Flex and APT). It can also create documents based on the input image structure, making one document per multipage image. Because our application is using the bank's own letter for document separation, we need to reconfigure this ruleset.

- CreateDocuments

  This task creates the document structure, based on the pages that you set up and configure. In our example, the package that the customer submits with the cover letter will be the first page of the document, followed by the first page of the attached bank statement, designated as Main_Page, followed by any additional pages of the bank statement, which will be designated as Trailing_Pages.

- Recognize Pages and Fields

  This runs a managed full-page OCR recognition of all pages (except attachments). In our application, everything is named CoverLetter, Main_Page, or Trailing_Page, so there will not be attachments designated in PageID.

- Fingerprint

  This task tries to match Main_Page to all known instances of Main_Page. If there was a previous example of a particular bank statement recorded, the Learning template matches against the existing fingerprint for that document so that we can use known zones for extracting the data. If there is not a previous example of a particular bank statement, the Learning template creates a new fingerprint automatically so that zones can be saved to the system for future encounters of statements from that bank.

- Locate

  Locate extracts data from zones if they are known for a particular statement and uses keyword searches or regular expressions to try to locate data if zones are unavailable or not found. This ruleset varies widely for different types of data that you need to capture, so in the template, we need to set this up.

- Lookup

  Lookup returns the fingerprint class of known fingerprints into the Fingerprint_Class field. In this application, we store fingerprints by the particular bank name that is associated with them. Therefore, this Fingerprint_Class field will hold the Institution, one of the date elements that we want to capture. Often, these types of values are in logos that are unrecognizable by OCR, so this is a more reliable way of populating institution names.

- Validate

  Like the Locate ruleset, this ruleset must be set up after the fields are added to the project. The purpose of the ruleset is to make sure that data conforms to the business rules. Every field should be validated in some way, to ensure compliance with data types and length restrictions in whatever system we are exporting to. Notice that Validate runs at least twice: Once during the Profiler task and again in the Verify task each time that an operator clicks Submit on a document.

- Routing

  This is used to prepare the documents for the Verify operator. Pages that violate a business rule are automatically shown to operators, but we need to also check that the data did not fall below the required confidence for the greatest accuracy, even though it might have passed a business rule. This ruleset is also commonly used to clean up any fields before they go to Verify. In our case, when the Lookup ruleset runs, it puts the `<New>` value in the Fingerprint_Class field when the lookup occurs, and we will be looking for that value and blanking it so that the operator does not have to erase before typing in the correct value.

- SetStatuses

  Different clients set rejection statuses in various ways. In our application, the rejection statuses are Delete, Rescan, and Review. Those statuses can be set directly in some but not all clients, so the Learning template also provides a drop-down menu to set them at Verify time. Regardless of which method an operator chooses to reject a document, this makes sure that the status is set correctly so that the rejected documents will not go to the repository and the appropriate people are notified.

- PreExport

  This is where the zones are saved to any `<New>` fingerprint that was created. Because it runs after Verify, we should now know where all of the required data is located on this particular type of bank statement. This ruleset is also usually configured to make the output image required by the system that we export to.

- Export

  This ruleset outputs data in a simple way to a text file. For production systems, Datacap offers many different types of export rulesets, and you can export to as many systems as you want.

- Process Exceptions

  This is where we handle any documents that were rejected by the operator (Delete, Rescan, Review). Customer requirements can vary widely, but typically this ruleset would be configured to export to some business management workflow so that appropriate action can be taken, but some customers just want notification so that documents can be fixed as necessary and inserted into future batches.

7.4 Building the document structure for the MktBankStmt application

The Learning template is used differently than the Forms template, in the fact that in the Forms template, we create a generic document and a generic page that had rules assigned to them to be inherited from other pages in the batch.

For flexibility, the Learning template is more conventional in the regard that you use the defined document and pages for the learning portion of your application, rather than creating your own and inheriting rules from them. You can add other page types, as we do in this application, but the Learning templates typically are not set up to inherit rules from the base Main_Page.

Because we use the bank letter as the separator page, we add that to our existing document in the template. However, we store all of the data on the Main_Page. When the bar code is read from the Cover_Page, we pull the bar code into a field on the Main_Page to do our lookup of the customer information. We also need to capture the Account Number, Date, and Balance from the bank statement starting on page 2 of our document.

Because we want to display the customer information (from a bar code lookup) to the data entry operator, we also need fields to hold those values. With some special work in a DCDesktop panel, we could just display the information as a text box or labels rather than have fields assigned to the customer information, but the field approach is fine. We will not be exporting that data however, because we already have it in the bank's databases. It is shown to the operator only so that they can verify that the name and addresses on the bank statement match the bank's values.

There are four fields already defined for the Learning template:

- Fingerprint_Class

  We use this field for the name of the bank that issued the bank statement. Whatever is in this field at export of a `<New>` fingerprint is considered its `fingerprint_class`, and looking at the fingerprints in Datacap Studio, you can see each bank that has submitted a statement and they are classified and sorted by class. This makes fingerprint maintenance much easier in the future. Remember that this fingerprint class is also used to populate the field when a bank statement matches a fingerprint that was added to the system previously.

- Index_Field1

  This is a generic fingerprint that is in the template as a placeholder to show you what rulesets at the minimum that you need to configure. Figure 7-3 shows the open event of Index_Field1 with rules in the Locate and Validate rulesets. You can see that there are only two rules currently attached to the field level, but that is a reminder that any fields that you add probably need field level rules in those rulesets also.

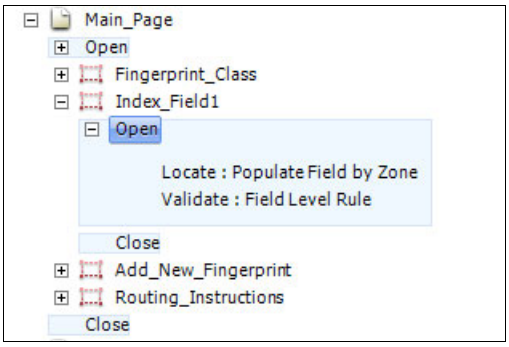


Figure 7-3 The open event of Index_Field1

- Add_New_Fingerprint

  Fingerprints are automatically added to applications using this template, as they are in Flex and APT. This field is represented as a drop-down menu on the Verify panel so that an operator can alert the system that the current document matched the wrong fingerprint. Because fingerprinting cannot be exact without having an extraordinary number of fingerprints in the system, there is a bit of fuzziness when matching. In time, two banks having statements in the system might use a similar layout, and when the second is added to the system, the application erroneously matches against an existing fingerprint from another bank.

If the operator sets this drop-down option to Yes, another fingerprint will be created in the system so you will have different and unique fingerprints for both banks. When fingerprint matching is done, the system gives the best match between the two. Under normal operation, you do not need to choose Yes in this box, even if it is a bank never before encountered by the system. If the system did not match another bank, it created a new fingerprint for this new layout automatically.

- Routing_Instructions

  Earlier, we mentioned exceptions such as Delete, Rescan, and Review. This field appears as a drop-down menu to the Verify operator if they need to route the document away from the normal export for special handling.

We can reuse Index_Field1 for one of our fields by renaming Index_Field1 in Datacap Studio or removing and replacing Index_Field1 in FastDoc. Due to the special nature of the Fingerprint_Class field, we cannot rename Fingerprint_Class without altering some of the rules, so we will not rename it.

Now, it's time to start editing the document structure in FastDoc. (Although we use FastDoc in this example, you can also use Datacap Studio to add and rename fields if you prefer.)

1. Add the new fields on the Main_Page. Figure 7-4 shows the FastDoc window with the fields added.

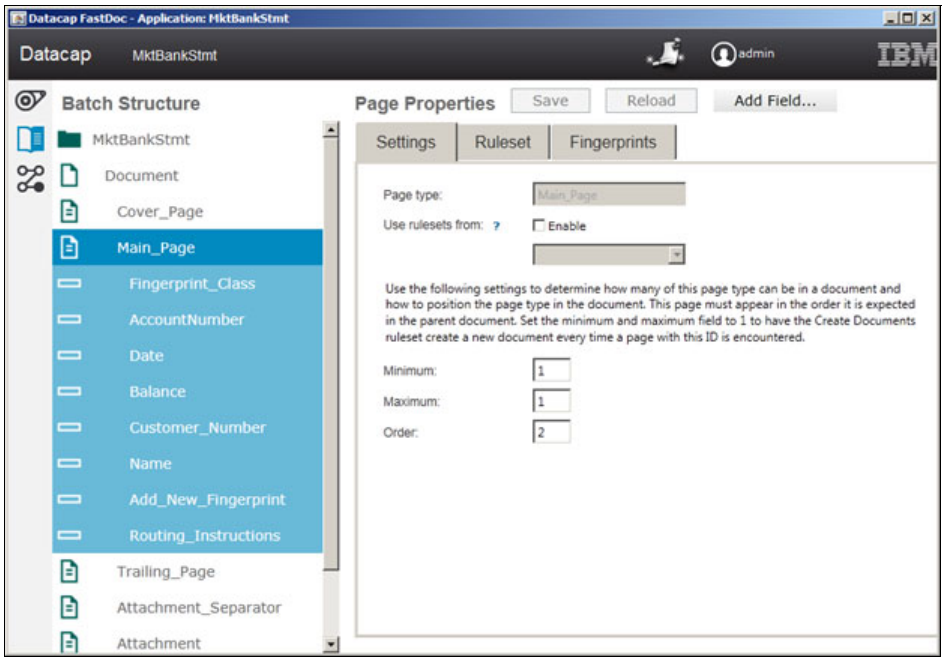


Figure 7-4 Fields added in FastDoc administrator mode

Notice also in Figure 7-4 that we started to set the document structure parameters by telling the system that the Main_Page will be the second page in the document, because we use the cover letter as the first page of the document.

2. Now, add the CoverLetter page. The order in the list does not matter, but the Minimum, Maximum, and Order variables all need to be set to **1** to say that the CoverLetter is not only required but also is to be the first page of the document, as shown in Figure 7-5.

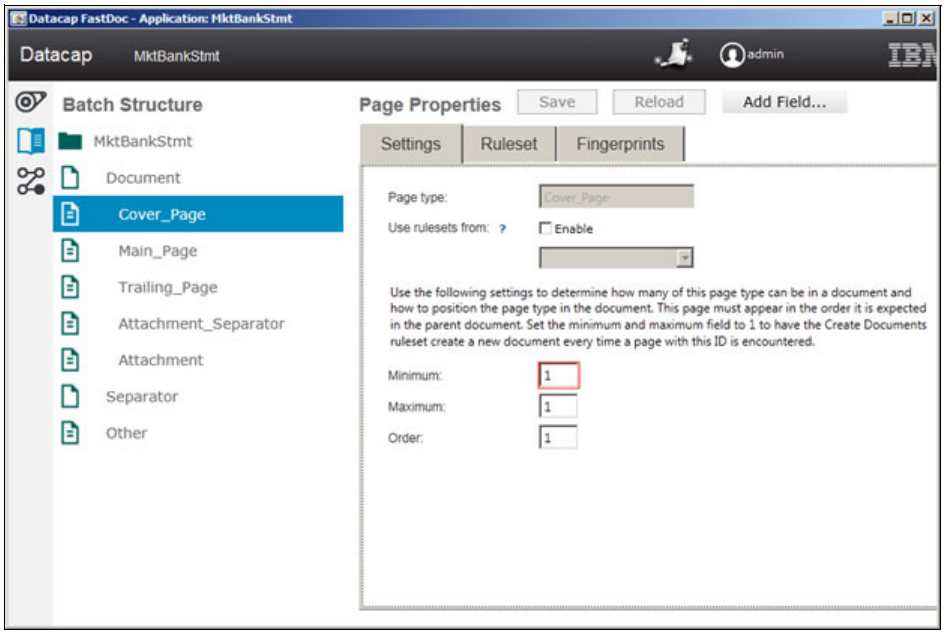


Figure 7-5 FastDoc administrator showing the Cover_Page addition

Before we actually configure the template to process batches of documents, we need to collect representative samples of a batch and put them in the proper folder for ingestion into the system.

For this application, we have six documents to use for our initial development. Because these documents are single-page TIFF files, they belong in the `\Datacap\MktBankStmt\Images\Input_SingleTIFFs` folder, as shown in Figure 7-6.

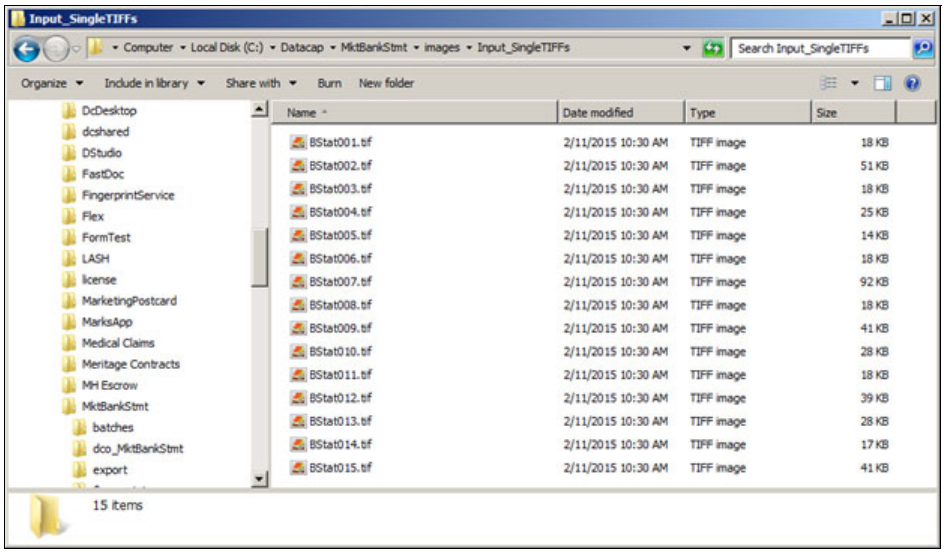


Figure 7-6 Placing the sample development documents in the proper folder

If your application contains images other than single page TIFFs, choose the `Input_MutiFormat` folder, instead.

Tip: You will probably run these same images over and over. To save time, consider running a batch and then pulling the single page TIFFs that result from the Convert Images task from the batch folder and using those single page TIFFs as your input for future tests.

You run many tests during your development time, and this will save you the time of having your images converted each time.

7.5 Configuring the Learning template for the MktBankStmt document structure

When you have good sample images in the proper directory, you should go through each ruleset and ensure that the ruleset is acting properly.

The VScan task should now work properly, but note that the ready-for-use configuration is for development only. Normally, when we input an image into the system, we move or delete the image from the input directory so that the image does not get processed again. When developing, however, we want to run our sample images over and over so to save time, we copy the images back into the input directory. When the project goes to quality assurance stage and production, you must set this up differently. There is a UI provided for the Input Files ruleset that should make this easy.

Similarly, the convert actions should already be set up for you in the template, and need no additional configuration.

The Managed Rotation ruleset is also preconfigured and normally does not need additional configuration. When images are input into a Datacap system, they are all set to be type Other, so this runs on every page. Because we have not done any Page ID yet when images reach this ruleset, so there is probably nothing that needs configuration here, either.

Image Enhancement is the first ruleset that needed configuration. Documents that can come from anywhere provide a variety of challenges, and most of them will be unknown at design time. Our sample images cover the basic challenges.

Tip: Be careful not to make too many adjustments. Any image enhancement that you set up here will operate on all images, and it is easy to tweak a particularly challenging image to perfection, only to have it adversely affect other images.

Primarily, we want to make sure that the image fix deletes lines that might interfere with later recognition, but not so aggressively that it obliterates our bar code on the Cover_Page.

At this time, it might be best to take a brief look at your images after they were enhanced by running a batch and looking at the .tif files in the batch directory for any obvious problems. Do not expect them all to be processed correctly. Now, we just try to focus on the Image Enhancement ruleset to make sure that the settings used are appropriate. The rest of the project has not been configured yet.

Open DCDesktop and run a batch through and look at each `.tif` image in the batch directory. If you are in doubt whether an image was able to be recognized correctly, you can look at the `.txt` file associated with that image to view the actual recognition results.

Again, try not to be too aggressive with the settings. When dealing with changing and unknown formats with virtually unlimited examples to come, determine whetherbalance to get the vast majority right. We expect that, occasionally, there will be images that can be enhanced better, but doing so might result in the loss of data in other images. There are no perfect settings that will do every image that you could ever encounter.

7.5.1 Configuring the PageID ruleset for the MktBankStmt application

The preconfigured rules in the Learning template look for a certain value on the separator sheets. The documents that we have though, do not have a static value. We need to reconfigure the PageID rules to process this batch properly.

We begin by seeing what we have to work with. From the batch we ran, we can open the `Profiler.xml` file to see how the default system detected the batch. Figure 7-7 shows an example of what we can expect to see when opening this file.

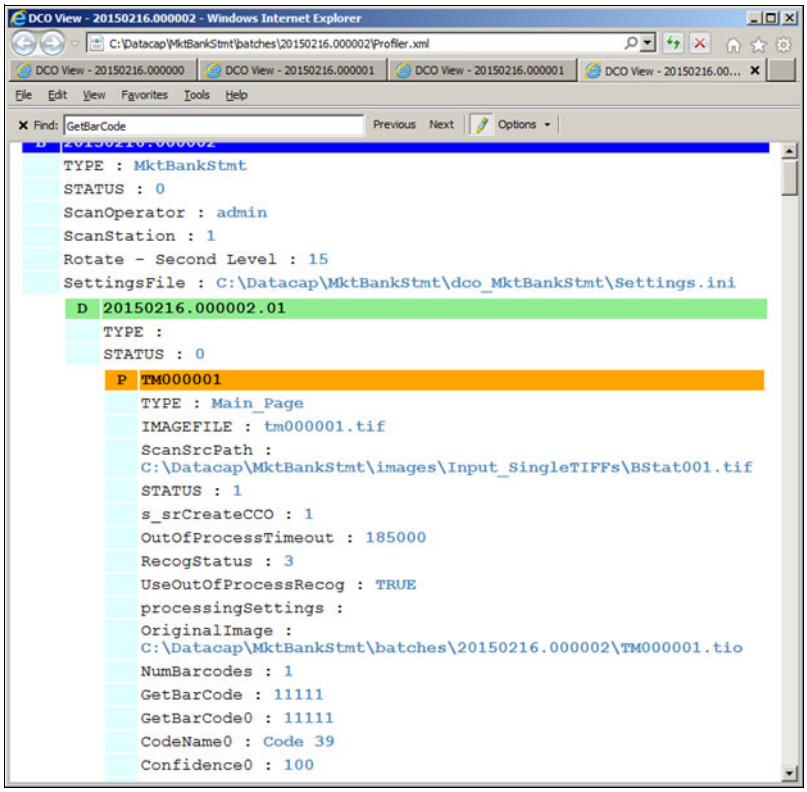


Figure 7-7 The profiler.xml file before configuring the PageID ruleset

Noteice that every page in the `profiler.xml` file is labeled as a Main_Page. Again, this is because the rules used in the Learning template are expecting a certain separator page, but we use something different. There is a variable on some pages called GetBarCode that has a value associated with it. These are our cover letters and we use them to determine the document structure.

To reconfigure the PageID ruleset, we go into Datacap Studio, lock the ruleset, and change the actions in there to look for pages that have a bar code on them.

Because we do not write an action to process the whole batch at once, we do not use the existing Batch Level Rule, so we must disconnect it from the batch level object, as shown in Figure 7-8.

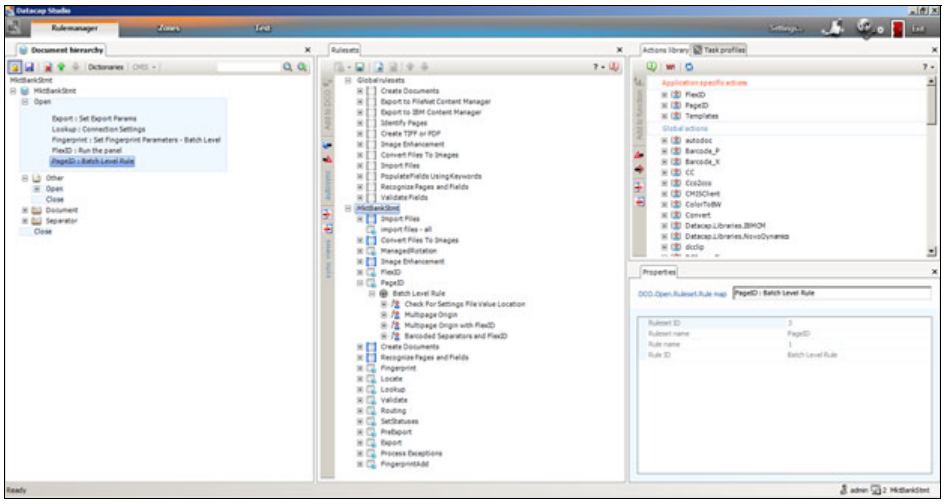


Figure 7-8 Locking the DCO and choosing to delete the rule binding

Because the rule is no longer bound and will not be used, you can delete it from the Rulesets panel.

We write the new rule to run on the page level. The first thing that we need to do is to detect our cover letters, and we can do this by checking to see if the *GetBarCode* variable contains a value, or rather, is not `NULL`.

To set the page type on the remaining pages in the document, we use `ChkLastDCOType` from the DCO rules library.

When completed, the rule should look as shown in Figure 7-9.

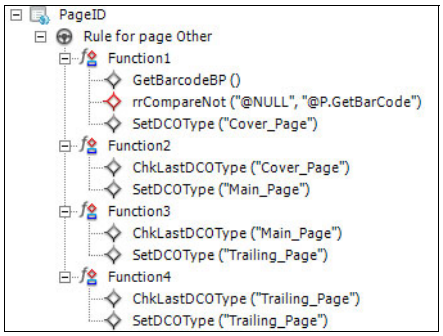


Figure 7-9 The newly configured PageID ruleset

The logic is that Function 1 looks for a bar code on the page and, if found, sets the page level variable *GetBarCode*. It then checks to see whether the *GetBarCode* variable on the page is not a null value. If it is null, the function fails and falls to the next function, where it will check to see what the previous page type was. If it was a Cover_Page, it sets the current page to Main_Page. If it was Main_Page, it sets it to Trailing_Page. Also, Trailing_Page types are set using Function 4. Each time a new Cover_Page is found, the scheme resets.

7.5.2 Bind the rule to the Other page type in the DCO

For a system going into production, this PageID method probably needs to be made a bit stronger. The issue will be found whenever a bank statement contains a bar code but was misidentified as a cover letter.

This shows the difference in getting a demo ready versus safeguarding the application for production. There are several ways to overcome this current application deficiency, and the better ones will involve some conversation with your customer.

The simplest method is for the batch preparation personnel to just draw a line through every bar code on a statement. This has some obvious downsides, because documents might be faxed or emailed, so no one will see them until they get to Verify and must be handled manually.

You could also pull the bar code into a field after fields are created, and then evaluate it with validation actions to make sure it is the proper data type and length.

Another option is to write a simple action to check the value of the variable. This is in the same vein as pulling it into a field, but without the extra actions and gymnastics within or code.

A suggested way in any case is to put some prefix onto each bar code, such as `XY`, so that you can check for the prefix, and then do a data type and length check on the remainder.

When the PageID is done, documents should be created correctly and the `CreateDocs` action should make the documents correctly.

`RecognizePagesAndFields` should do full page recognition on the images in a managed fashion.

`FindFingerprint` should also work from the template with no modifications. It does fingerprint matching on only the Main_Page, which you'll recall is the first page of the bank statement following the cover letter. If there are new fingerprints, they get created automatically and stored in the `<New>` fingerprint class, and these new fingerprints carry a new TemplateID value. Otherwise, if these new fingerprints match an existing fingerprint, they carry the TemplateID value of the fingerprint that they best matched against.

The next ruleset to configure is the Locate ruleset.

7.5.3 Configuring the Locate ruleset for the MktBankStmt application

Now that you know what you want to capture, you need to decide how to capture it. There are four ways of getting information from an image:

- Zonal
- Regular expression
- Keywords
- Operator entry

We need to decide how to get each piece of information from all the different bank statements that we will encounter (thousands of them, all different formats). Because this is a LearningTemplate application, we have two tasks:

- Learn how to find information about documents that we have never encountered before.
- Learn how to capture the data for documents that we have encountered before.

Table 7-1 lists the fields that we need to capture with the methods that we want to use to capture them for this example.

Table 7-1 Fields and method of capture

Field to capture	Method for first encounter	Method thereafter
Fingerprint_Class	Operator Entry	Database Lookup from Fingerprint database
AccountNmber	Keyword	Zonal, failing to keyword
Date	Keyword	Zonal, failing to keyword
Balance	Keyword	Keyword, failing to Zonal
Customer_Number	Read from Cover_Letter bar code	Read from Cover_Letter bar code
Name	Database lookup from Customer_Number field	Database lookup from `Customer_Number` field

At the document level of the Locate ruleset in the template, the CCO files for Main_Page and Trailing_Page are merged so that all of the data from the bank statement is in one easy-to-search place.

At the page level of Main_Page, the zones are read in, if they exist. Remember, zones only exist on bank statements that have made through the export process and the positions saved in an `FPXML` file. If an `FPXML` file does not exist with the zonal information with the field, the position on the fields remains 0,0,0,0.

Now, try to capture the fields. The `Fingerprint_Class`, `Routing_Instructions`, and `Add_New_Fingerprint` fields should be set up to work for you already, by virtue of using the LearningTemplate. For the `Fingerprint_Class` field, The operator must enter the data on the first encounter, but it should automatically populate thereafter when a similar bank statement is found. The `Add_New_Fingperrint` and `Routing_Instructions` fields are simply drop-down menus, and we set their default values.

First, we work to get the bar code from the Cover_Letter onto the `Customer_Number` field on the Main_Page. There are several ways to do this, but the easiest is to just copy the bar code value from the Cover_Letter page to a document variable, and then fill the `Customer_Number` field on the Main_Page with that value.

Use `rrSet(@P.GetBarCode, @D.CustomerNumber)` to copy the bar code value to the document level into a variable called *`Customer_Number`*. When the following page runs (Main_Page), the document-level variable can be copied into the field by using `rrSet(@D.CustomerNumber,@F)`.

Figure 7-10 shows the two rules added to the Locate ruleset and bound to the appropriate DCO objects.

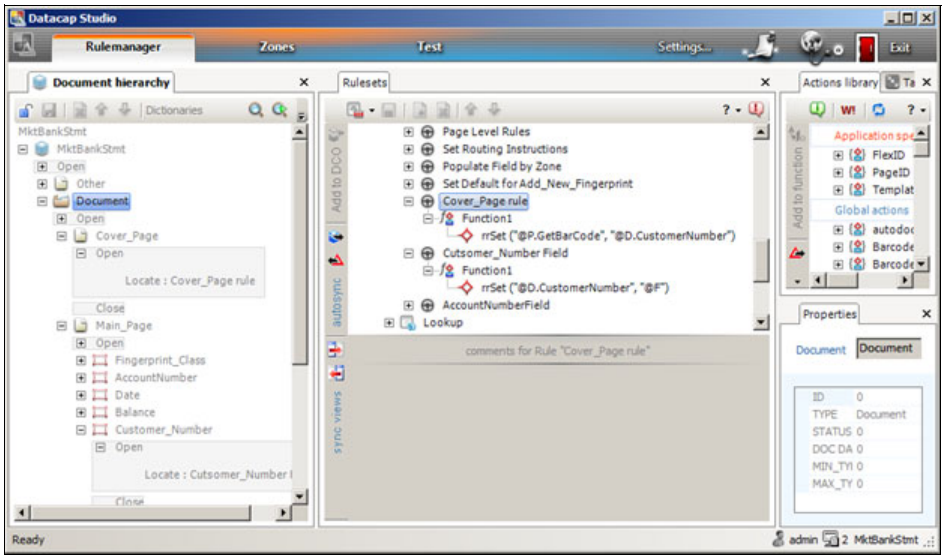


Figure 7-10 Capturing the Customer_Number into a field on Main_Page

You'll build the rulesets for AccountNumber and Date nearly identically.

When you use the `PopulateZNField` action, if there is no zone configured for the data. it fails, and the trailing function begins to execute. So it is safe to look for data zonally even on the first encounter of the fingerprint, because if no zone exists yet, the action runs rules to find the data programmatically by searching the merged CCO.

Therefore, this should be the first function on these two fields:

```
PopulateZNField
```

If there was a zone for a field and if there was data found at that location, the data is pulled into the field and there is no further searching. The trailing functions can then look for data programmatically.

In our example we want to use a keyword search:

```
FindKeyList(Keyfile name)
GoRightWord(1)
Check the data type, depending on which field it is
Updatefield
```

Figure 7-11 shows the Locate rule that is bound to Account_Number to capture the field.

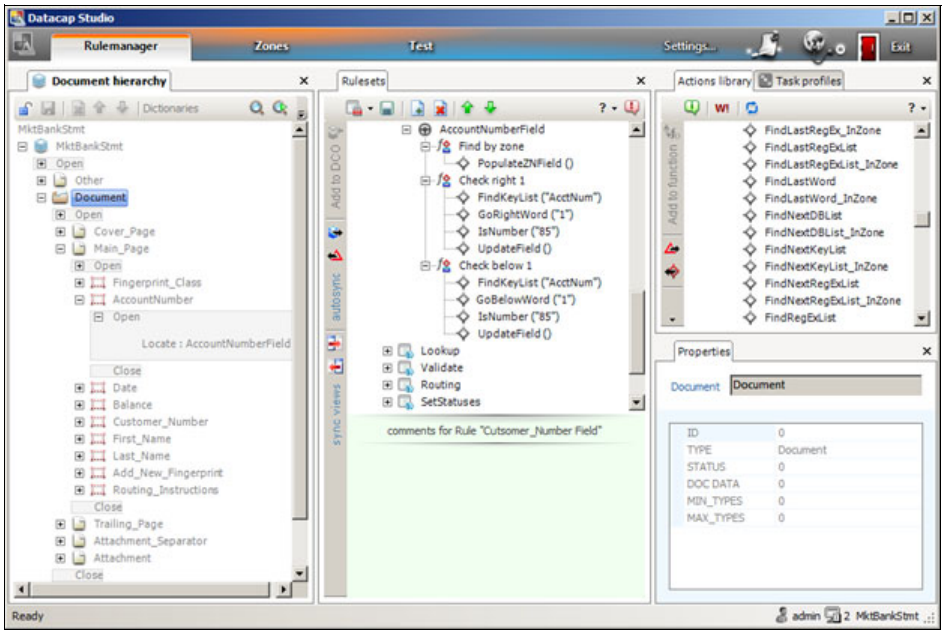


Figure 7-11 The AccountNumber field configured and bound for capture

Next, we make a key file that contains a list of the labels that you want to search for, in the order that you want to find them.

We create a new text file called `AcctNum.Key` in the `dco_MktBankStmt` directory and enter the following values:

```
Account Number:
Primary Account:
Primary Account
Account number:
PREMIER PLUS CHECKING
```

As new words or phrases are encountered as labels in future statements, users can add to this file, and these labels will be checked when programmatically searching for the data. Order your list from most-specific to least-specific to get the best matches. For instance, “Primary Account:” needs to appear in the list before “Primary Account” (without the colon). As FineKeyList searches, if it looked for the value without the colon first, it would match and never check for the one with a colon. Similarly, if you find a statement in the future that just says “Account” with the number to the right, it would be best to add it toward the bottom of the list so that it will check for the more specific “Account Number” and “Account number:” first.

With this structure, we search the CCO for each word in the key file until we find one. The search is through the entire document for the first word, and then the entire document for the second keyword, and so on. You can use `AggregateKeyList` to search the CCO for all words at one time, so be careful about adding non-specific types of labels, such as just “Account” if you are using `AggregateKeyList`. If a statement has “Your Account Summary” as a title, it would never search down low enough to get your account number.

You have to balance. Too far one way you will match incorrectly to the wrong word, but too far the other and you will miss some values that might have been found. With the Learning template, it is best to be toward the later end of the spectrum to perhaps miss some of the less-encountered words and phrases, because after this document has gone through Verify and the operator has provided a location for this value on a particular banking statement, the system will use that zone for future encounters with this type of statement rather than a keyword match.

Going through the other logic of the action, if the keyword is found, we go to the next word to the right and check its data type. If there is no word to the right, or if whatever is to the right is the wrong data type, it fails, and we can check a different direction from the found keyword.

If everything is OK when we get the value to the right of the keyword, we update the value of the field, and we are finished.

If it fails, we try something similar, but perhaps looking below or two words to the right. With rules, you can search all around a found keyword for your proper data type.

With the `Date` field, we use the same structure by copying the rule from AccountNumber and just make the changes required:

1. Make a new keyword file, with the labels based on the date.
2. Change the data check to `IsDateValue()`.
3. Be able to group the words for the date and test it.

   A date such as July 4 2015 normally just goes to the word *July*. `GroupWordsRight(1.5)` groups any words in vertical proximity (within 1.5 spaces) to the right of first word found. This is shown in Figure 7-12. It is nearly identical to the AccountNumber rule, so you can begin with a copy.

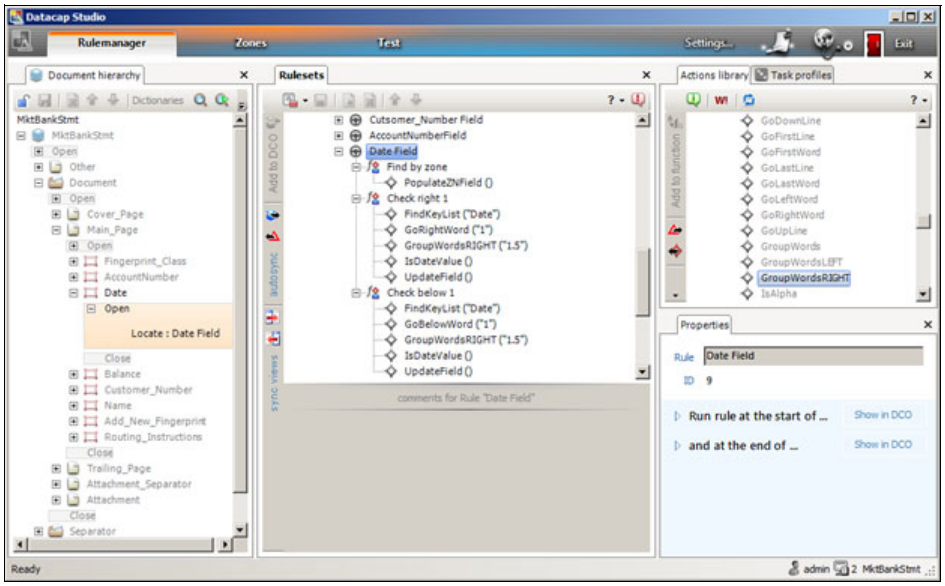


Figure 7-12 The Date rule bound to the Date field

Balance is similar. However, because totals on these types of documents tend to float around, depending on the activity of the account, do the keyword searches first. If the search fails, use the zone from the first encounter. As before, you can start by copying the AccountNumber rule and modifying it to look at a different keyword file and check for a different data type.

Figure 7-13 shows the Balance field rule. Notice that the `PopulateZNField` is the third function on this field, not the first.

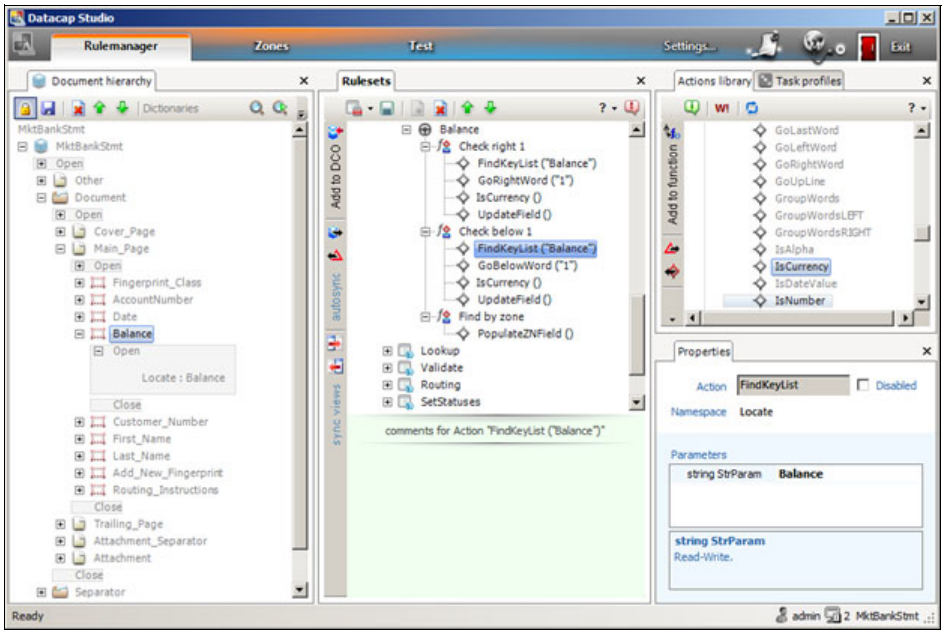


Figure 7-13 Configuring and binding the Balance rule

We also populate the Name field before it goes to a data entry operator, but we add that function in another ruleset.

7.5.4 Performing the database lookup on CustomerNumber

Because we will have the data in the Locate ruleset by the time the First_Name and Last_Name fields are processed, we can simply configure lookups in Locate. However, in this example, we choose to make a new ruleset to handle the calls into the customer table to make the code easier to locate in case the lookup ever needs to change (if the organization adds a stored procedure in the database to return values, for example).

Also, most people prefer to open the connection to the database, do all of the lookups, and then close the database. So isolating calls to that particular database allows the connection to take less time if the ruleset is dedicated to looking up data in that database.

We set up our connection in two parts:

- One runs in Batch Profiler and fill the fields with values if the bar code was successfully read.
- One is performed only at Verify time if a Verify operator corrects a misidentified Cover_Letter.

The background lookup in Batch Profiler opens the database connection on the open event of the batch-level object. Do the lookup on the Name field to populate it.

For the batch-open rule in the new ruleset, we copied the batch level rule from the Lookup ruleset. Remember, this ruleset looks up our Fingerprint_Class based on the matched fingerprint to provide, in this case, the bank name from the statement.

We copied this ruleset is because of a built-in enhancement. If you log in to the application with a station ID that ends with `-Test`, the ruleset can programmatically call a different database connection than if you log in without it. When you are disconnected from the corporate LAN where the corporate database resided, and you work offsite, it is nice to be able to configure a connection to a local version of the database, perhaps on your notebook, to continue working on the system. As configured in the template, however, the ruleset points to the same connection. You must alter `-Test` to get this function.

On the field level, we want to perform the actual lookup. SmartSQL enables you to embed smart parameters directly in your SQL string. Therefore, on the first image, the following statement:

```
SELECT Name from Customer where CustNum = '+@P\Customer_Number+';
```

Will evaluate to:

```
SELECT Name from Customer where CustNum = '11111';
```

Lastly, on the batch close event, we want to close the database connection. Look closely at Figure 7-14 to notice the new ruleset, where each rule is bound to the DCO and where the ruleset is placed in the Batch Profiler task profile. The new ruleset shows bindings on the left, the rules and actions in the middle, and placement in the task profile on the right.

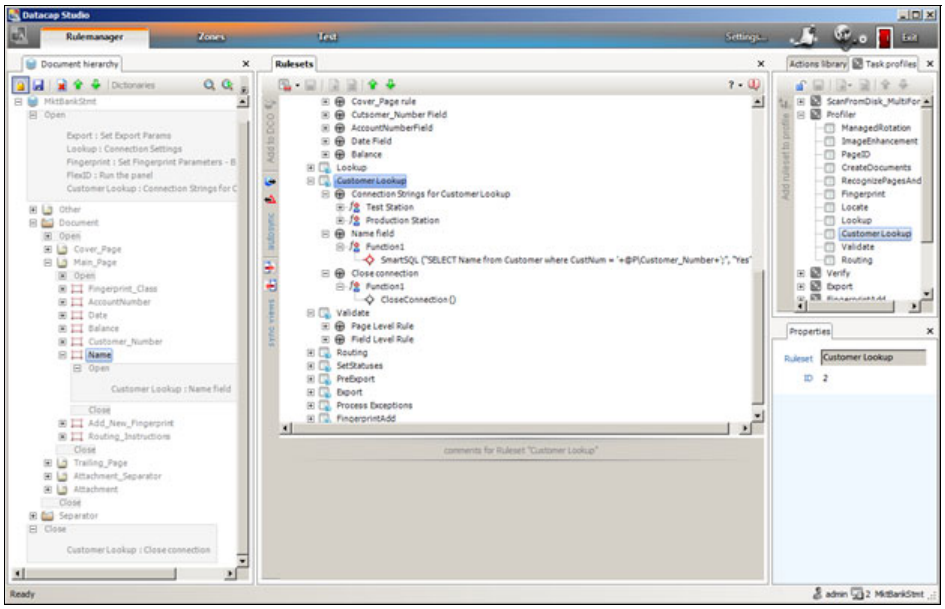


Figure 7-14 The new ruleset

As mentioned, we also want to configure the Name field to be able to populate a value from the database at Verify time, and this is done by placing a lookup variable on the field.

You need to lock the DCO and right-click the **Name** field to access the **Manage Variables** menu, as shown in Figure 7-15.

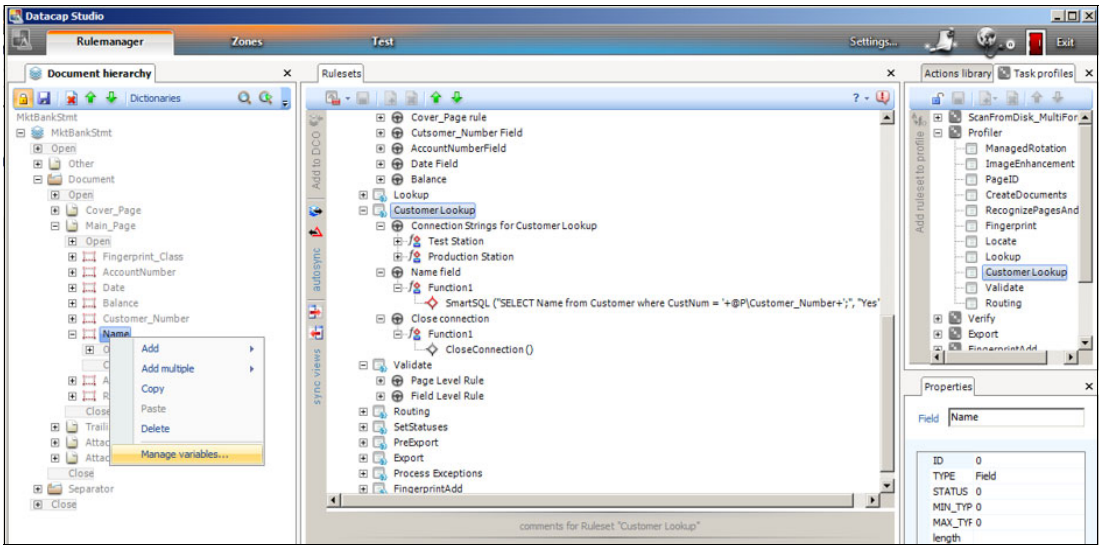


Figure 7-15 Accessing the Manage Variables menu

From here, click **ADD** to add a new variable named *Lookup* with the following value:

```
<SQL flist='Customer_Number,Name' dsn="*/lookupdb:cs">SELECT CustNum,Name FROM
Customer WHERE CustNum= '@@Customer_Number@@'</SQL>
```

Figure 7-16 shows adding the *Lookup* variable so that the Verify operators can perform lookups with the hypertext link on the Verify panels.

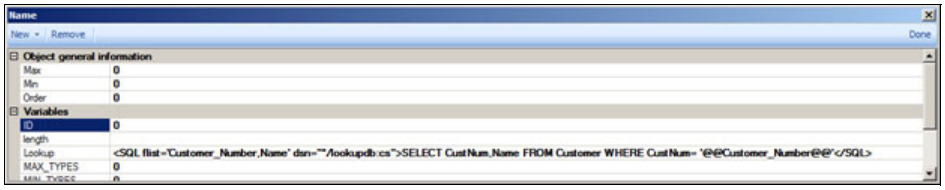


Figure 7-16 Adding the Lookup variable

The syntax for this is a bit arcane, but the `flist =` lists the fields in the order that a returning record set will populate, the `dsn` is the connection string, and the last section is the SQL statement, with field names encapsulated in `@@` symbols to denote that we want the field values inserted there.

7.5.5 Performing validations

In practice, every field should be either constrained to specific values or validated in the validation ruleset so that you can be sure at export time that everything is the correct data type, format, and size to meet you export requirements. It is important to avoid letting a data entry operator pass a value that will abort the export process.

The validation rules typically clean your data of unwanted characters, reformat the data if necessary, and perform all checks so that, programmatically, you know that the data will not pass unless it can be exported.

The `AllowOnlyCharacters` action is typically used to clean the data from unwanted characters. There are validation actions to remove or replace specific characters, but `AllowOnlyCharacters` expects only characters that you want to keep, not remove, so it is usually easier to implement and maintain.

Table 7-2 shows how we want every field that is not constrained by a drop-down list or database lookup to be validated.

Table 7-2 Documenting the validations

Field to validate	Cleaning	Validations
Fingerprint_Class	None.	Maximum length of 36 characters. Cannot be blank.
AccountNmber	Remove all but digits and dashes.	Maximum length of 24. Minimum Length of 5 if filled. Might be blank.
Date	Replace with / (slash).	Valid, within the last 60 days, and formatted to YYYY/MM/DD
Balance	Remove all but digits and decimal point.	Currency value. Cannot be blank (minimum length of 4).
Customer_Number	Constrained by bar code value.	No additional required.
Name	Constrained by DB lookup.	No additional required.

Most of the above is obvious, but the cleaning part of the Date field might need some explanation. The date validation routines currently do not work with dates separated by dashes, such as 7-4-2015, but they do work with 7/4/2014. Because we do not control the format of he date on the form, we must replace the dashes with slashes to perform the validation.

Also, you might be curious about why we do not clean the Date field further, possibly to only digits and separator characters. This is because our checking and conversion actions can take in textual month names and convert them properly. We do not want to filter those out.

The last thing we need to consider is whether the rules that we impose are overridable. Things such as data types typically are not, but lengths and character sets generally are. Ask yourself: What would we do if the data on a form genuinely did not match a business rule? For non-overridable rules, the data entry operators will be stuck on that page unless they delete it or mark it for review. Some organizations might want this feature, but others will not.

Figure 7-17 shows our validation ruleset with the validations from our chart in place. Notice on the date that one portion of the rule is overridable, but the other is not (to prevent sending an invalid date to the export system).

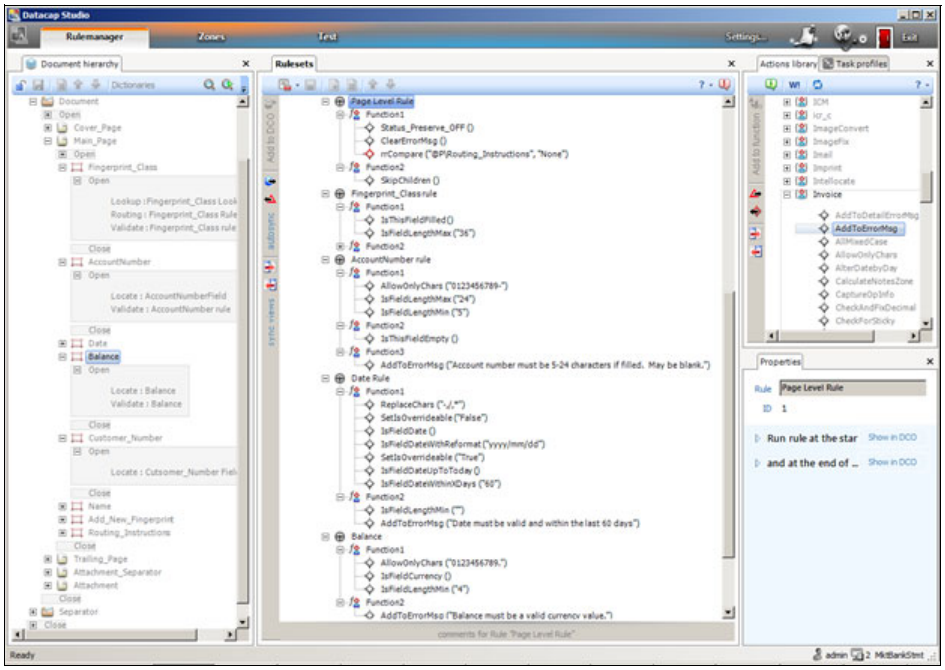


Figure 7-17 The validation rules and their binding

7.6 The routing ruleset

In the Learning template, Datacap displays to the operator only pages that have either a validation or a confidence problem. This is the default setup.

There is an additional rule that runs on the Fingerprint Class field on new images input into the system. Rather than displaying `<New>` as the bank name, it clears the field and causes the page to show up in validations. To this rule, for this application, we also added an action to set the label variable on the field. When the label variable (all lowercase) is present, the verification panels should display that label rather than the default field name.

7.7 Testing with a validation panel

After you have your rules set up, you can start running batches through. Look at them in the validation panel, and also type fake values to test your validation rules. At this point in the process, you might find that your application needs some enhancements.

We run batches through every time we change something substantial in a step. See Figure 7-18 on page 169 for an example of what the test batch should look like in Datacap Desktop running the default panel.

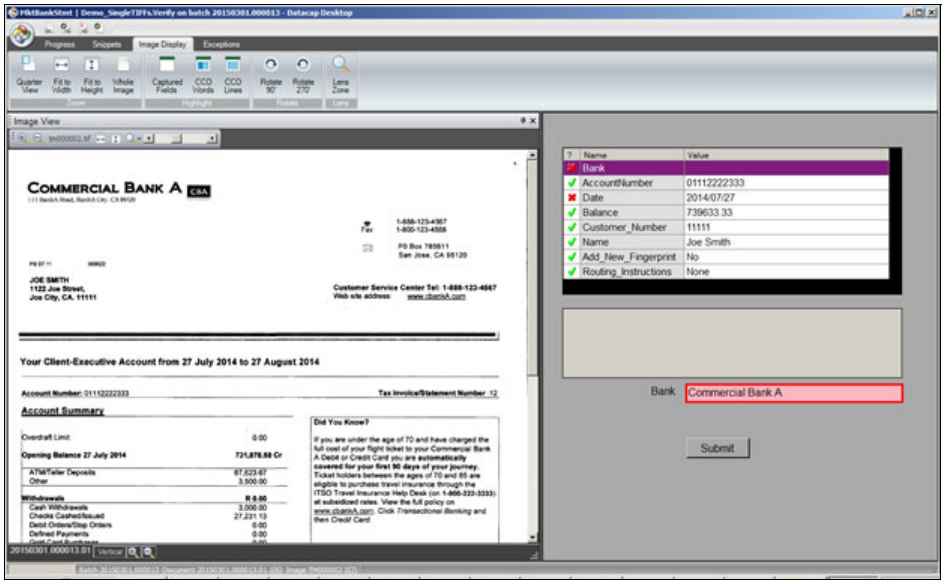


Figure 7-18 The MktBankStmt application in Datacap Desktop with the default panel

After the batches are through the Verify step, we have just a bit more to do before we wrap this up.

7.8 Exporting

One of the first things that happens in the `Export` task profile is a process called *Intellocate*. It moves the `<New>` fingerprints to a classification according to a field value, in our case, the field value of the Fingerprint_Class field. Remember, this value is also read in when we are processing bank statements, so if it is a statement type we have seen before, we do not have to identify the bank. Intellocate sets the fingerprint classification according to the value typed into the Fingerprint_Class field when the statement type is initially processed.

The results after Intellocate should look like Figure 7-19.

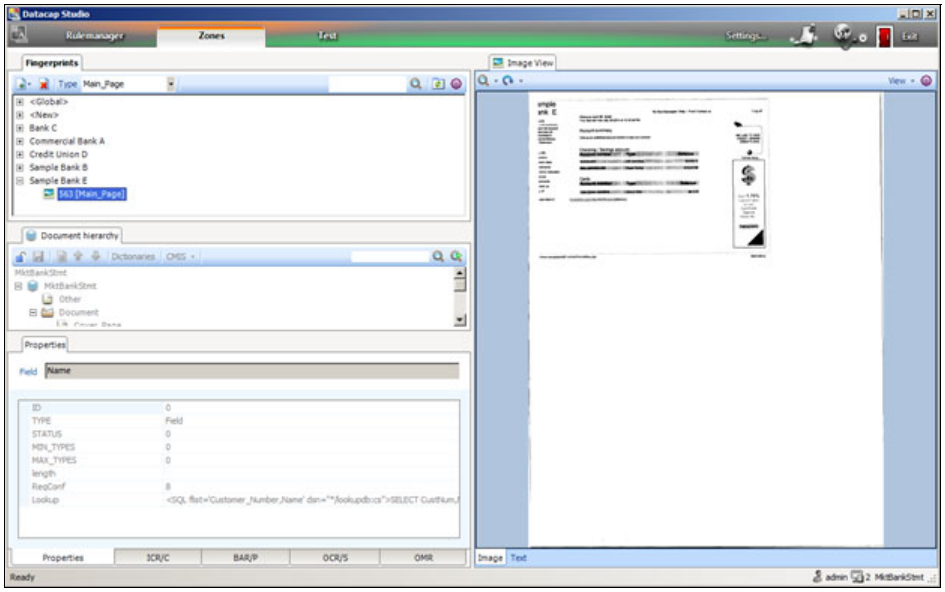


Figure 7-19 Intellocate sets the fingerprint classification

Intellocate also writes the zones where the data was extracted from to the `FPXML` file. As a result, the application can now process the bank statement types that it has seen before without as much operator intervention, because the application can use zones in the future.

You can export data from Datacap to many systems. This process is easy with correctly validated data. Many systems have a UI ruleset to make configuration even easier. For more information about exporting data, see Chapter 9, “Export and integration” on page 205.

7.9 Wrapping up your project

The project is not finished yet. We need to figure out some way to handle documents that have something other than the value of None in the Routing_Instructions field, because normally, they are not exported.

Your customer might want to customize a Datacap Navigator or Datacap Desktop panel to show the results with a different look and feel. For more detail about Datacap Navigator, see Chapter 10, “Datacap user experience in IBM Content Navigator” on page 233.

Some customers might want you to set up an email system to email particular parties if routing is needed, others might have you export to a special system such as BPM to let operators handle the exceptions in another product. Remember that these exception documents should not be exported to the normal data and image repositories. Therefore, it is important to deal with the exception documents in some manner so that the batch does not end with documents that have not been dealt with according to what the customer wants.

You should also set up Datacap Report Viewer for the reporting of statistics that are automatically collected, and also Datacap Maintenance Manager to automatically delete old batches, and to notify someone and reset batches that are not progressing through the system for one reason or another.

Using the Learning template will give you a solid foundation for creating applications dealing with unstructured documents, and do it in a standardized way that is documented and understood by other developers and support personnel. Building on this foundation will make unstructured application development much faster and less challenging than in the past.

8

System scalability, availability, backup, and recovery

This chapter describes how to design your capture system for high availability, scalability across multiple applications and ingestion routes, performance, backup, and recovery options and strategies for IBM Datacap.

This chapter includes the following sections:

- System scalability, performance, and availability
- Datacap Rulerunner
- Backing up and restoring
- Configuring Datacap Rulerunner
- Multiple Ingestion routes
- Interfacing Applications

8.1 System scalability, performance, and availability

Building an enterprise solution across multiple locations with large user numbers and a high volume of document throughput requires a system that can scale both horizontally and vertically. This system must also be able to withstand one or more node failures.

IBM Production Imaging Edition is made up of two key components: IBM Datacap to scan and capture images and IBM FileNet, which is the central repository to store these images. Both Datacap and FileNet have several options to design a scalable and available system. This chapter concentrates on scaling the Datacap component of Production Imaging Edition.

For information about scaling the FileNet component, see the IBM Redbooks publication titled *IBM FileNet P8 Platform and Architecture*, SG24-7667, which explains the approaches and methods of scaling, performance tuning, and availability.

8.1.1 Typical Datacap installation

A typical Datacap implementation for a small scale project consists of the following servers:

- The Datacap server, which is the central service that provides user authentication, workflow and queuing.
- A Datacap Rulerunner Server that runs tasks that do not require human interaction, such as optical character recognition (OCR), intelligent character recognition (ICR), optical mark recognition (OMR), bar code recognition, and export to repository. This is typically the most process-intensive server, and hardware should be optimized for use.
- The Datacap Web Server that runs the web-based Admin and User Interface.
- The Datacap Web Services to enable use of both the IBM Content Navigator user interface and the mobile app. This web service can also be used to interface with other systems, such as multifunction devices (MFDs) and Smart Network Scanners. This service can run on either Microsoft Internet Information Services (IIS) or as a standard Microsoft Windows service.
- The WebSphere Application Server, which hosts the IBM Content Navigator instance. Content Navigator is designed to be the future replacement of the current Datacap Web Server UI for admin and verification tasks.

Figure 8-1 shows a simple IBM Datacap architecture.

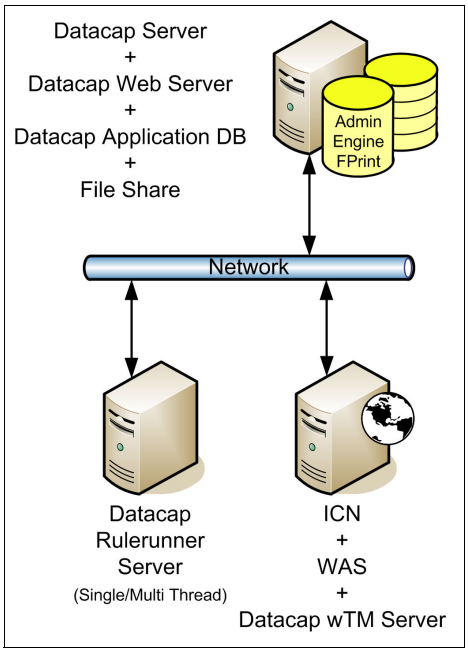


Figure 8-1 A simple IBM Datacap architecture

Although it is possible to run all Datacap components on the same physical machine, use this architecture only for the smallest production capture systems and for development configurations. Performance is limited because Datacap Rulerunner tasks are typically processor-intensive and can interfere with Datacap server response times.

8.1.2 Scaling Datacap Rulerunner Server vertically (scale up)

Datacap Rulerunner Server use can be scaled vertically. You can do this in the following ways to handle the volume of documents and images that you want to process:

- Increase the hardware specification of the Datacap Rulerunner Server so that processes can run faster. To realize this goal, you can perform such tasks as upgrading processors, increasing memory, increase disk speeds (for the central batch file system), and using a faster network connection.
- Increase the number of available Datacap Rulerunner threads per server to allow multiple tasks to process concurrently and use multi-core processors more efficiently. For more information about this process, see 8.2, “Datacap Rulerunner” on page 190.

Multi-thread licensing: At the time of publication, an additional licenses, called the *Datacap Rulerunner PVU license* or the *Datacap Enterprise UVU license*, is required to permit use of multithreading in Datacap Rulerunner. Contact your IBM sales representative for more details.

There is a limit to how much you can scale the hardware configuration of a single Datacap Rulerunner Server. The limit can be a physical limit. For example, you cannot get faster components. You can also reach a point at which the benefits of upgrading hardware are less, such as in terms of cost versus performance, than purchasing of a new separate machine.

Datacap Rulerunner Server is a 32-bit application and, therefore, is restricted to 2-4 GB of addressable memory. However, Datacap Rulerunner Server creates separate processes for OCR and other intensive tasks, which allows for use of additional memory outside of this range if the operating system is 64 bit.

Vertical scalability relies on adding processing power to a physical machine. Configuring a system with just one Datacap Rulerunner Server implies a single point of failure for background processes.

8.1.3 Scaling Datacap Rulerunner Server horizontally (scale out)

Datacap Rulerunner Server use can be scaled horizontally by increasing the number of servers used to handle a high volume of documents or images.

In an installation of this type, you have one or more centralized Datacap servers and a defined number of Datacap Rulerunner Servers for unattended processing of ingested documents. Depending on your license, you can configure these servers to run a single thread or use multiple threads. For more information, see 8.2, “Datacap Rulerunner” on page 190.

Figure 8-2 shows a typical horizontal configuration, with a centralized Datacap server and three single-threaded Datacap Rulerunner Servers that share the load of document processing. It also shows other key server components that are separated.

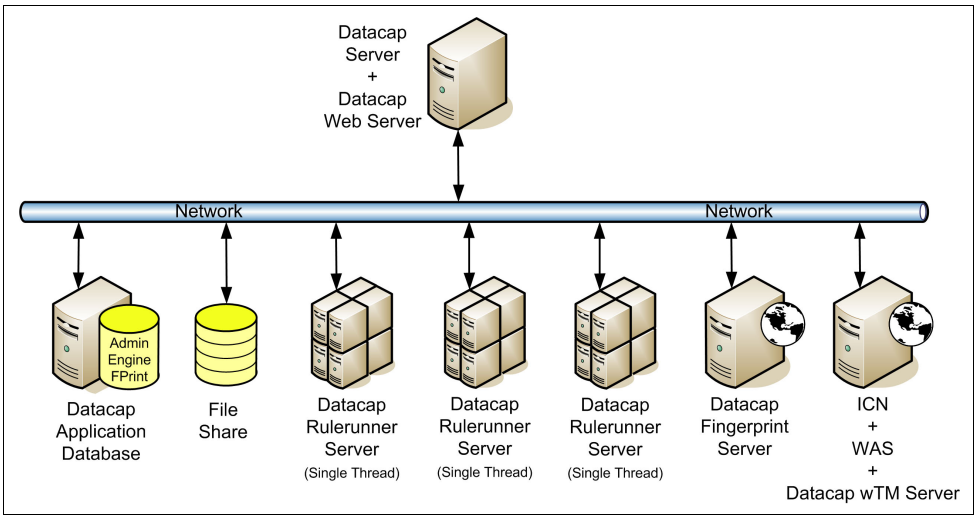


Figure 8-2 A horizontal configuration of Datacap Rulerunner Server

Each Datacap Rulerunner Server uses the first-in first-out (FIFO) algorithm to poll the Datacap server for the oldest batch of documents pending for a set of defined background tasks. Then the Datacap Rulerunner Server processes each batch in turn.

Datacap Rulerunner can be configured to poll for specific tasks in given projects. To scale effectively, tasks must be shared intelligently across the available Datacap Rulerunner Servers. For most purposes, it is best to configure all Datacap Rulerunner Servers to run all background tasks to ensure that any work is processed by the first available Datacap Rulerunner Server as soon as possible. If a Datacap Rulerunner Server fails, the remaining Datacap Rulerunner Servers continue to process all tasks.

Alternatively, you can assign specific subsets of work to a subset of Datacap Rulerunner Servers. This method helps primarily to control the amount of processing power assigned to specific parts of the workflow. For example, it ensures that priority is given to certain tasks, even if older work is available for other tasks.

A trade-off between performance, cost, and resiliency must be determined on a case-by-case basis.

8.1.4 Scaling Datacap Rulerunner Server horizontally and vertically

IBM Datacap Rulerunner Server can be scaled horizontally and vertically to get the advantages of both methods:

1. Scale up by upgrading the hardware specification of the Datacap Rulerunner Servers. This increases single-server throughput by maximizing hardware use on the server.
2. Scale out by adding additional Datacap Rulerunner Servers.
3. Increase the number of processing threads that are available to each Datacap Rulerunner Server by upgrading to the Rulerunner Enterprise license.

Figure 8-3 illustrates a horizontally and vertically scaled IBM Datacap implementation. It details the use of many multi-threaded Datacap Rulerunner Servers. It also shows use of a separate database server, file share, and dedicated Datacap Fingerprint Service. More information about these components is provided later in this chapter.

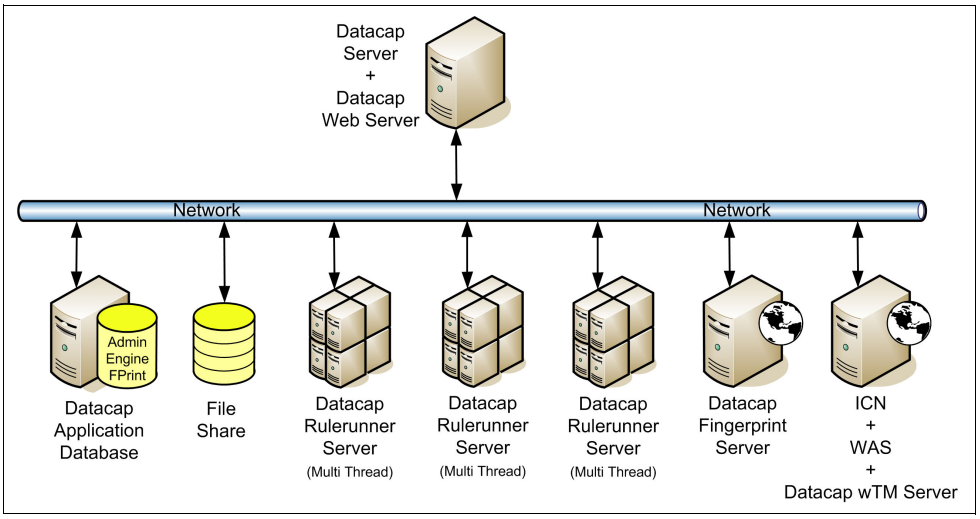


Figure 8-3 A horizontal and vertical Datacap Rulerunner Server configuration

With this approach, you have the advantages is that you remove single points of failure for processing tasks and maximize the performance of available hardware.

For a walkthrough of a Datacap Rulerunner configuration, see 8.3, “Configuring Datacap Rulerunner” on page 196.

8.1.5 Datacap server scaling and redundancy

As explained in the previous sections, you can scale Datacap Rulerunner horizontally and vertically. You can achieve this scaling by increasing the number of Datacap Rulerunner Servers, increasing the hardware specification, and increasing the number of processing threads by the addition of a Datacap Rulerunner Enterprise or PVU license. This approach is suitable for task processing, such as OCR or validation. However, all of the batches that are being processed are managed by the Datacap server.

The Datacap server is the central Windows service. It provides user authentication, workflow, and queuing (and file services for Datacap Web Server and Datacap wTM Server). Without this core component, no tasks can be managed and updated in the system.

Typically, the load on a Datacap server is low. Processor and I/O-intensive processes, such as OCR and Export, are run on the Datacap Rulerunner Servers. The main function of Datacap server in this scenario is to access the Datacap queuing database (the Engine database) to select batches for processing and update statistics. Roughly six database transactions are performed for each batch. As a result, moderate loads can be supported easily with a single Datacap server. The load must be quite high to require use of an additional server. Significant improvements are made in Datacap 9 to allow for greater Datacap server scalability.

However, there are exceptions. For example, when using Datacap Web Server to permit the use of web clients, an increased load is placed on the server to transfer batch files to and from the web server. The same is true for use of Datacap wTM and, therefore, the Datacap Navigator UI (IBM Content Navigator) or mobile app. For more information, see 8.1.7, “Datacap Web Server scaling and redundancy” on page 181.

Also, using the Report Viewer tool increases the load on Datacap server to transfer data from the database to the IIS Web Server. If large data sets are transferred, this added load can be significant and might necessitate scale out of Datacap servers.

If the hardware limitation of the Datacap server is reached, in the first instance, the server hardware can be upgraded to meet the demand. If the server hardware does not meet the requirements, additional Datacap servers can be installed. Datacap Rulerunner Servers, Datacap Web and Datacap wTM can be configured to access the additional Datacap servers, as needed.

In high availability (HA) environments, the suggested practice is to have two or more Datacap servers to permit failover and balancing.

Active-active configuration

Figure 8-4 shows two or more Datacap servers that are configured as companion servers, but with separate IP addresses and server names.

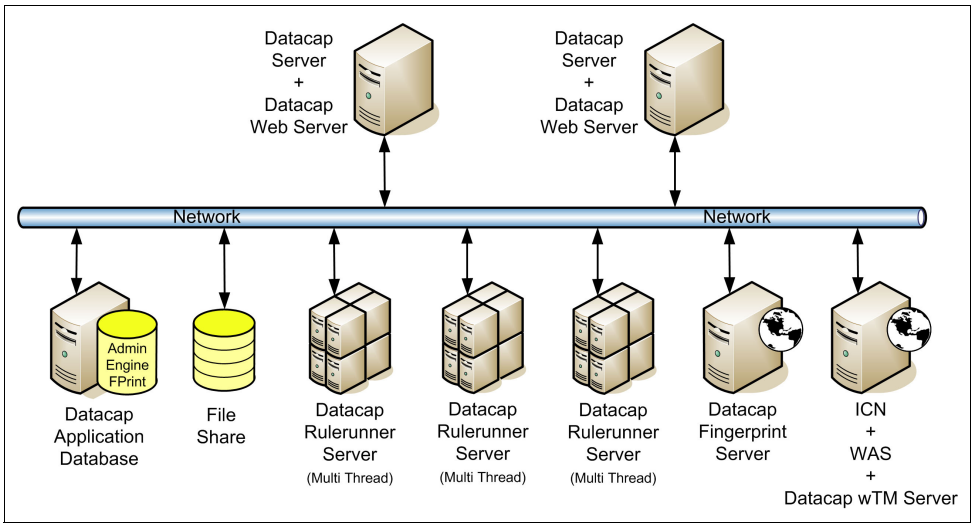


Figure 8-4 Datacap servers configured for Active-Active

The Datacap Rulerunner Servers (single-threaded or multi-threaded) can poll either or both Datacap servers separately, looking for tasks to process. Load balancing of Datacap Rulerunner Servers against Datacap servers is achieved by smart Application and Task configuration and is covered in 8.1.4, “Scaling Datacap Rulerunner Server horizontally and vertically” on page 177.

Use of a load balancer appliance between two or more Datacap servers permits load balancing of Datacap Clients such as Datacap Desktop. Datacap wTM Server and Datacap Web Server can also be configured providing each TCP/IP session is set as “Sticky.” This is not a necessity for balancing Datacap Rulerunner Server, although it can be used.

As of this writing, for Datacap 9, all batch creation tasks must be configured or operated so that they can access a single Datacap server. Conflicts between multiple Datacap servers creating new batches can result in skipped batch IDs and delays in batch creation on some servers.

In this configuration, if one or more Datacap servers fail, the clients will disconnect and then connect to the remaining Datacap servers. However, each Datacap Rulerunner Server must be configured accordingly to achieve this connection.

The Datacap servers use the same database servers and Universal Naming Convention (UNC) path so that the same project can be shared across multiple Datacap servers.

Active-passive configuration

In an active-passive configuration, a single Datacap server is configured as the primary server. A second Datacap server is configured on a secondary server with the same parameters as the primary server, meaning they share the same IP address and server name.

If the primary Datacap server fails, the secondary server is started manually. Because its configuration is identical to the failed server, it begins to process in the same manner as the primary server. All clients and Datacap Rulerunner Servers connect as though they are the primary server.

Batch abandonment and rollback

In the active-active and active-passive configurations, if a Datacap server fails, any tasks that are in process by clients connected to that server (Datacap Desktop, wTM, Datacap Web Server) are abandoned in the running state. The secondary server needs to reset such batches before reprocessing.

Batch reset, also known as *rollback*, can be performed manually or through the use of the Datacap Maintenance Manager. In certain cases, you must restore the former state of the batch for proper reprocessing. This topic is beyond the scope of this book and is not covered here.

8.1.6 Scaling both Datacap and Datacap Rulerunner servers

Figure 8-5 shows a possible configuration of Datacap servers and Datacap Rulerunner Servers in a scaled high availability configuration. In this example, we have omitted any additional clients to simplify the scenario and concentrated on the Datacap server and Datacap Rulerunner Server components.

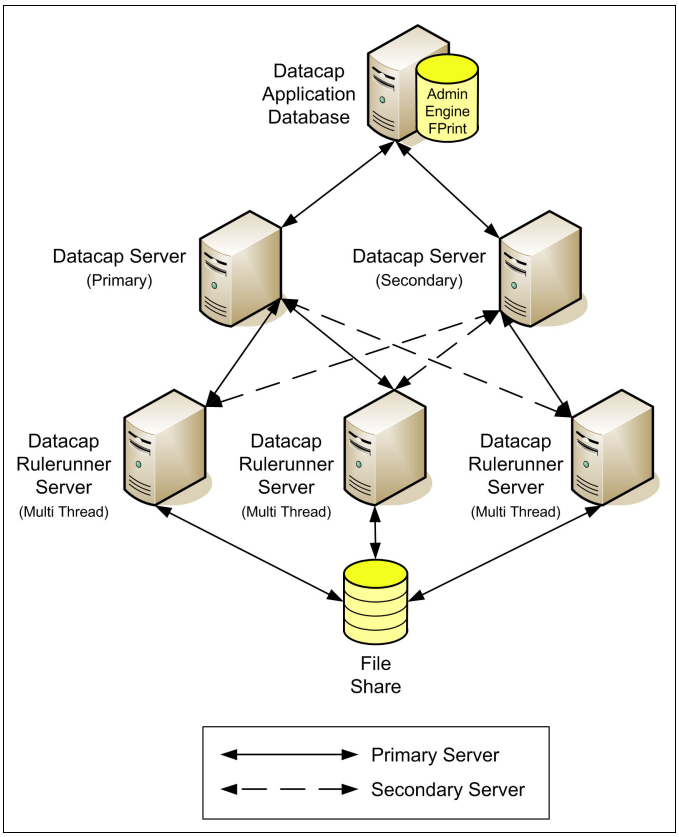


Figure 8-5 Scaled high availability configuration of core components

The two Datacap servers are connected to a single database server. With this connection, both servers can serve out tasks and batches from the same application, and each server updates the batch status to the common database. If one Datacap server fails, the other Datacap server still has access to the database and batch status. This configuration can then be scaled out with more Datacap servers as needed. However, consideration must be given to the load of the database server also.

In this configuration, each Datacap Rulerunner Server (single-threaded or multi-threaded) has been configured to poll one primary Datacap server considerably more than the other (primary higher priority server is polled more frequently). However, it occasionally polls the secondary Datacap server (secondary lower priority server). The fact that Datacap Rulerunner Server is aware of both Datacap servers means that, if one fails (the primary high priority server in this example), it will start using its secondary lower priority Datacap server. It drops its priority and polls the secondary Datacap server for all allocated tasks it has configured to process.

Taking this approach, you can balance the load of the Datacap Rulerunner Server and configure it for high availability without the need to include a dedicated network load balancer. However, the load balancer can still be used, if required.

This scenario can then be built out to accommodate more Datacap Rulerunner Server and more Datacap servers as needed.

8.1.7 Datacap Web Server scaling and redundancy

Although the Datacap Web Server is superseded by IBM Content Navigator UI or Datacap Navigator, it is still possible to use this UI.

Datacap Web Server relies on the use of Microsoft Internet Information Services (IIS) to serve out a web-based user interface for administration, scanning, and data verification tasks. Because IIS might reside outside of a firewall, for security purposes, all file and database requests are routed through the Datacap server. This routing ensures that all such requests are run securely within the firewall. It also places a greater load on Datacap server than the Datacap Desktop client.

Datacap Web Server response times depend on the Datacap server performance. When many users run tasks with large amounts of I/O, such as image upload and database lookup, implementations, including Datacap Web Server, must pay attention to the specification and configuration of Datacap machines to ensure adequate performance.

Rules such as validation and document integrity, which are triggered during Verify tasks, are also run on the IIS under Datacap Web Server. Heavy use of validation or other rules also increases the load on the IIS.

If the number of concurrent users who are using the Datacap Web Server exceeds the capability of IIS, additional IIS servers with Datacap Web Server configured pointing to the Datacap servers should be added. Use of a load balancer can be used here. Client Browsers connect to this load balancer. The load balancer redirects the requests to individual IIS servers by using round-robin scheduling, or other defined method. The Datacap Web Server uses session cookies, so you must configure the load balancer to persist sessions based on the client's IP address. Typical capacity range from 50 to 100 users for each instance of IIS for use of the Datacap Web Server.

If the throughput generated by Datacap Web Server is too great for a single Datacap server, you can add, separate Datacap servers in an active-active configuration. In a similar way a load balancer can be added. The individual IIS servers connect to this load balancer. The load balancer redirects the requests to individual Datacap servers by using round-robin scheduling or other defined method.

As a guide, the typical capacity range from 50 to 100 users for each instance of IIS for use of the Datacap Web Server.

Figure 8-6 shows several load balanced Datacap servers and load balanced IIS Datacap Web Servers providing UI access to several Datacap web clients.

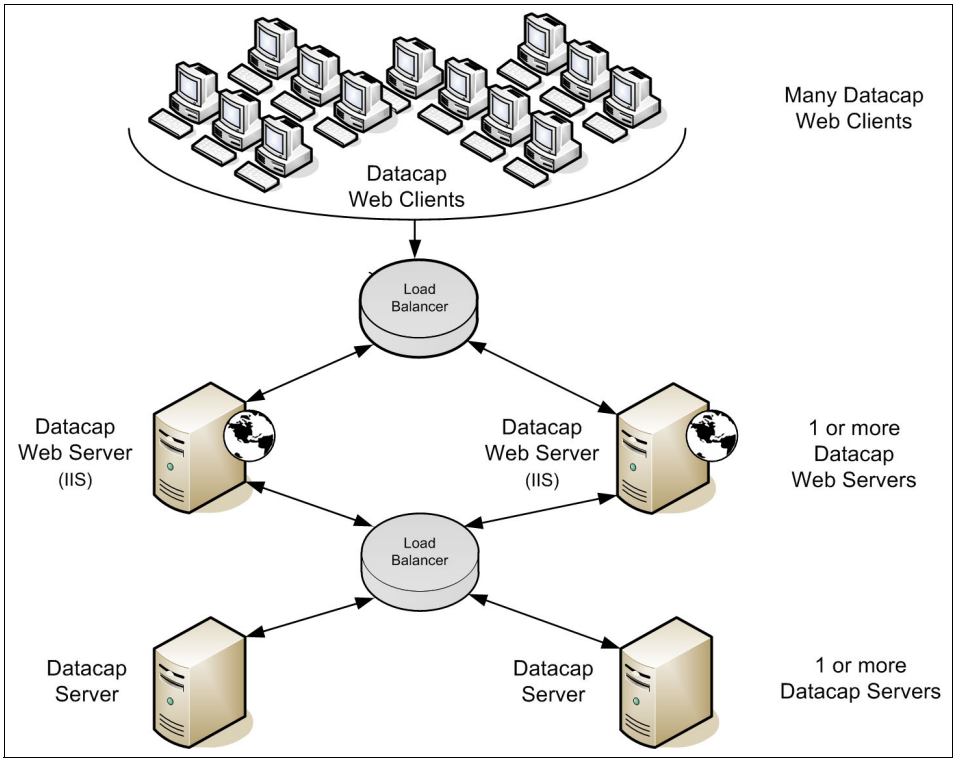


Figure 8-6 Load balanced Datacap servers and Datacap Web Servers

8.1.8 Datacap wTM Server scaling and redundancy

IBM Datacap Web Services is the REST-based web service software component of IBM Datacap. This provides the ability to interact with the system through a simple, platform-independent, application programming interface (API).

wTM is a Microsoft Internet Information Services (IIS) based web service that can be installed on a dedicated web server, or can, in smaller implementations, be installed on a web server on which other Datacap components are installed. wTM can also be configured to run as a Windows Service negating the need of IIS. IIS is still required for use of the Fingerprint Service.

The wTM API supports `HTTP GET`, `POST`, and `PUT` methods that allow you to create a new batch, upload pages to the batch, set the page file name, update page files, release a batch to the next task, to retrieve any file in the batch folder (including image files), retrieve batch information such as batch ID and batch status, run rules, and perform admin tasks.

wTM is the service that delivers the integration point for IBM Content Navigator, Datacap Mobile apps and certain third-party MFD integrators. This API allows them to connect to the Datacap system using a standardized approach.

In a manner similar to Datacap Web Server, wTM can be configured to scale by using a hardware load balancer.

Figure 8-7 shows a possible configuration for distributed MFDs in a load balanced and high availability environment.

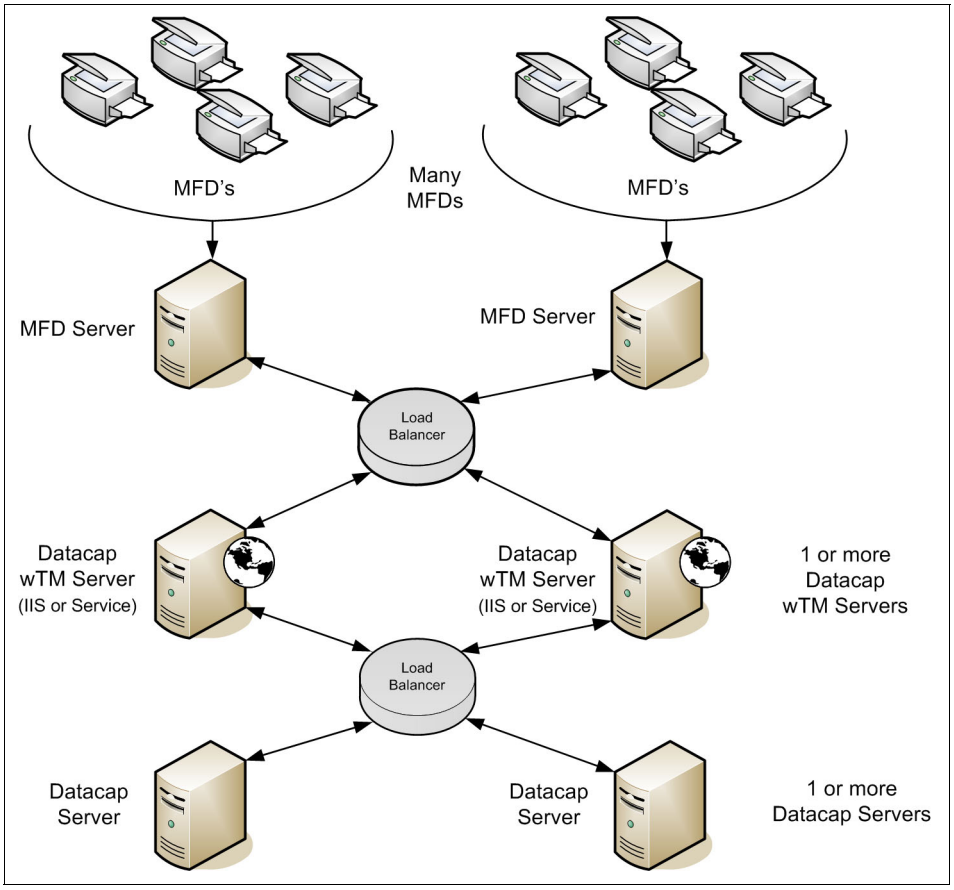


Figure 8-7 Example of a load balanced configuration for MFD integration

Figure 8-8 shows a typical wTM configuration for use of Datacap Navigator web clients. One or more wTM Servers are load balanced against one or more WebSphere Application Server and IBM Content Navigator servers.

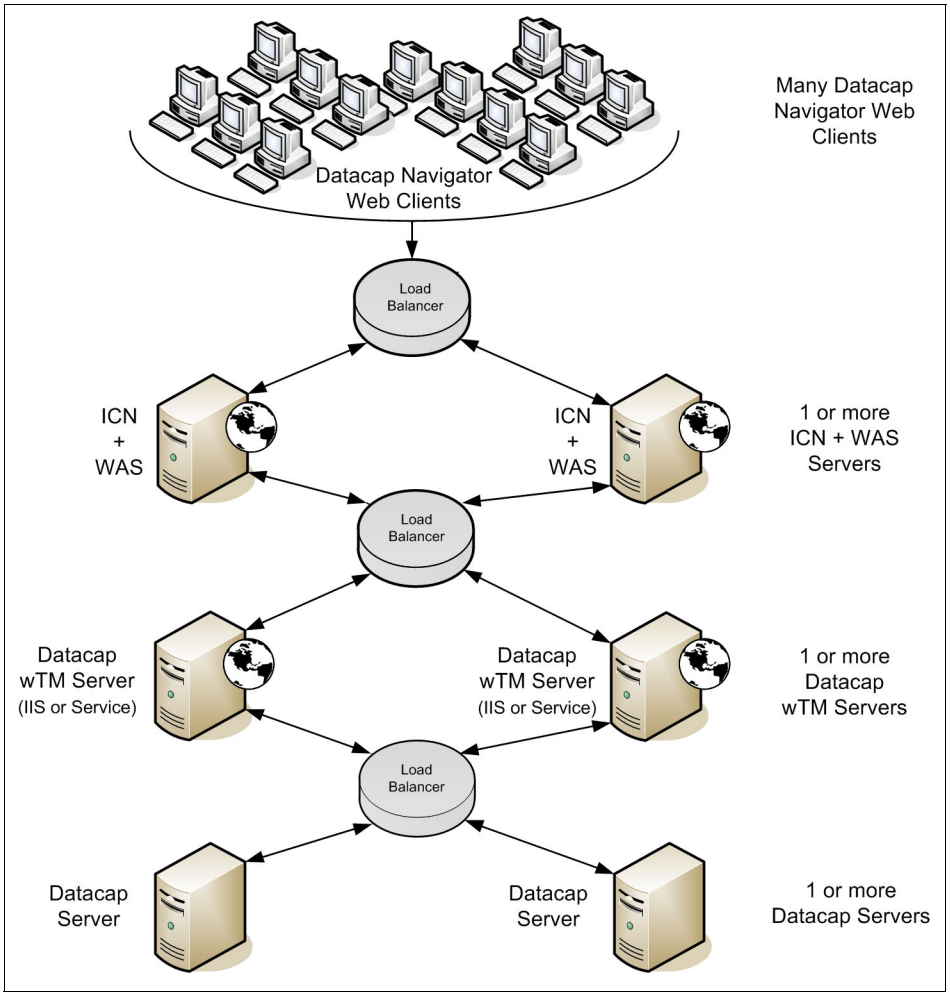


Figure 8-8 Datacap Navigator load balanced across multiple WebSphere Application Server servers and Datacap wTM servers

Figure 8-9 shows how multiple mobile capture clients can be load balanced against one or more Datacap wTM Servers.

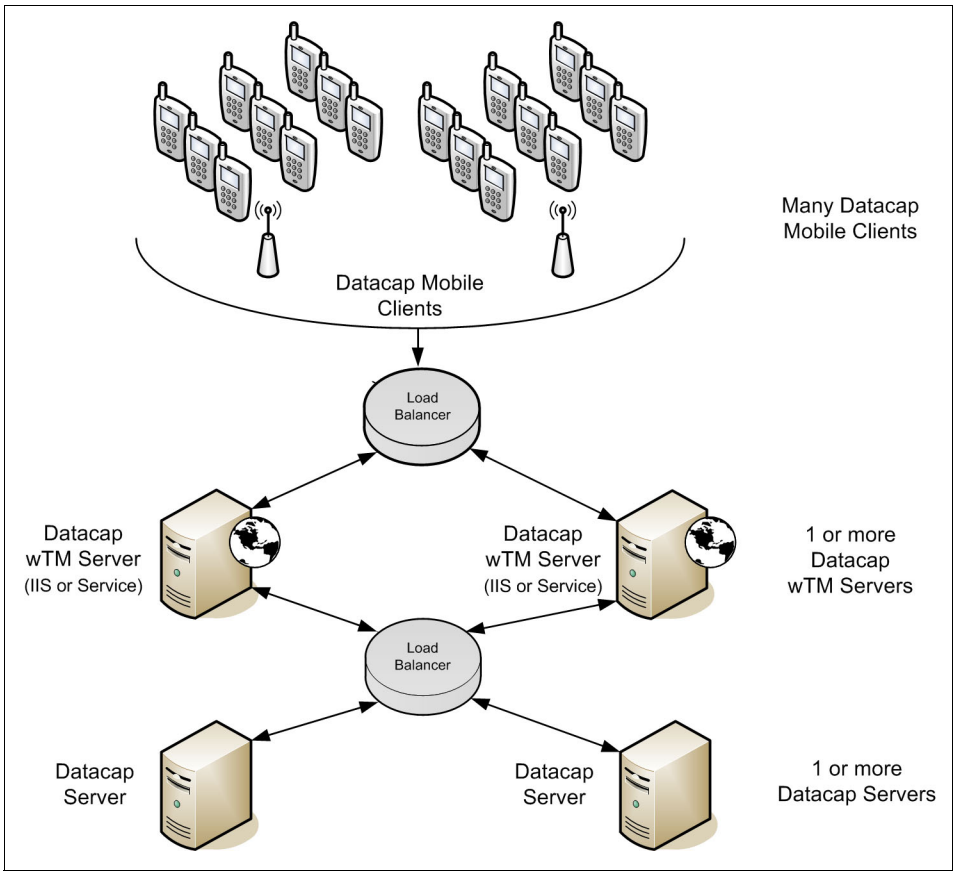


Figure 8-9 Load balancing for multiple mobile capture clients

wTM uses session cookies, the load balancer should be configured to persist sessions based on the client's IP address. If a server fails, users who are connected to the failed server receives an error message and must log in again.

Considerations must be taken into account when load balancing wTM. In large-scale implementation wTM should be located on its own server, ideally local to the clients it serves. Appropriate bandwidth must be provided between Datacap server, file share, and the interacting client, that is, IBM Content Navigator, mobile app, or MFD server.

In a way that is similar to Datacap Web Server, all file and database requests are routed through the Datacap server. This routing ensures that all such requests are run securely within the firewall. It also places a greater load on Datacap server than when using the Datacap Desktop client.

wTM response times depend on the Datacap server performance. When many users run tasks with large amounts of I/O, such as image upload and database lookup, attention must be paid to the specification and configuration of Datacap server machines to ensure adequate performance.

Rules such as validation and document integrity, which are triggered during Verify tasks, are also run on the wTM Server. Heavy use of validation or other rules also increases the load on the IIS or Windows Service running wTM.

If the number of wTM Server concurrent users exceeds the capability of the host machine, additional servers with wTM that point to the Datacap servers should be added.

8.1.9 IBM Datacap Navigator scaling and redundancy

The Datacap Navigator Client is installed within IBM Content Navigator by way of a Java plug-in provided as part of the core IBM Datacap installation.

The Datacap Navigator Client is built on IBM Content Navigator technology, which is hosted on IBM WebSphere Application Server.

For the BM Datacap Navigator Client to function, it requires connectivity to Datacap wTM Server. wTM is described in 8.1.8, “Datacap wTM Server scaling and redundancy” on page 182.

Figure 8-8 on page 184 shows a typical scaled Datacap implementation of Datacap Navigator. Here Datacap Navigator web clients are load balanced against one or more instances of IBM Content Navigator. IBM Content Navigator and WebSphere Application Server are in turn load balanced against wTM servers.

The same consideration as described for scaling wTM should be adhered here to ensure optimal configuration between Datacap Navigator and wTM.

For more information about installing and configuring, see the IBm Content Navigator documentation in the IBM Knowledge Center:

www.ibm.com/support/knowledgecenter/SSEUEX_2.0.3/contentnavigator_2.0.3.htm

8.1.10 Datacap Desktop scaling and redundancy

Datacap Desktop clients connect to the Datacap server to authenticate, start batches, verify pages, perform administration, or invoke several other tasks that are not automated.

A Datacap Desktop client connects to the primary Datacap server as defined in the `.app` file of the application to which it is trying to connect. If this Datacap server fails, the client connection drops.

To reinitiate a connection, the client must be restarted. Upon restarting the client again, it first tries to connect to the primary Datacap server. If this server is still down, the client then attempts to connect to a secondary Datacap server (if configured).

It is also possible to define which Datacap server a client connects to as a primary server. This approach can help to scale out with additional thick clients beyond the capacity of a single Datacap server.

Figure 8-10 on page 187 shows the configuration of Datacap Desktop clients connecting to a primary Datacap server. It also shows the redundancy to connect to a secondary Datacap server.

Typically the central `.app` file is used for configuring the use of the Datacap servers. However a custom local version can be used to define a local failover configuration different from the central version.

It is important to note that Datacap servers can be load balanced using a hardware network balancer if wanted.

A key difference between the Datacap Desktop client and the web-based clients is that it connects directly to the file share of the Datacap system. This removes the need for any interim servers between the client and the actual batch files. As a result, the speed of execution is faster than use of web-based clients.

For example, its use in high-speed scanning environments out-performs the web-based interfaces. However, the location of the client and the file share need to be considered when designing a capture system.

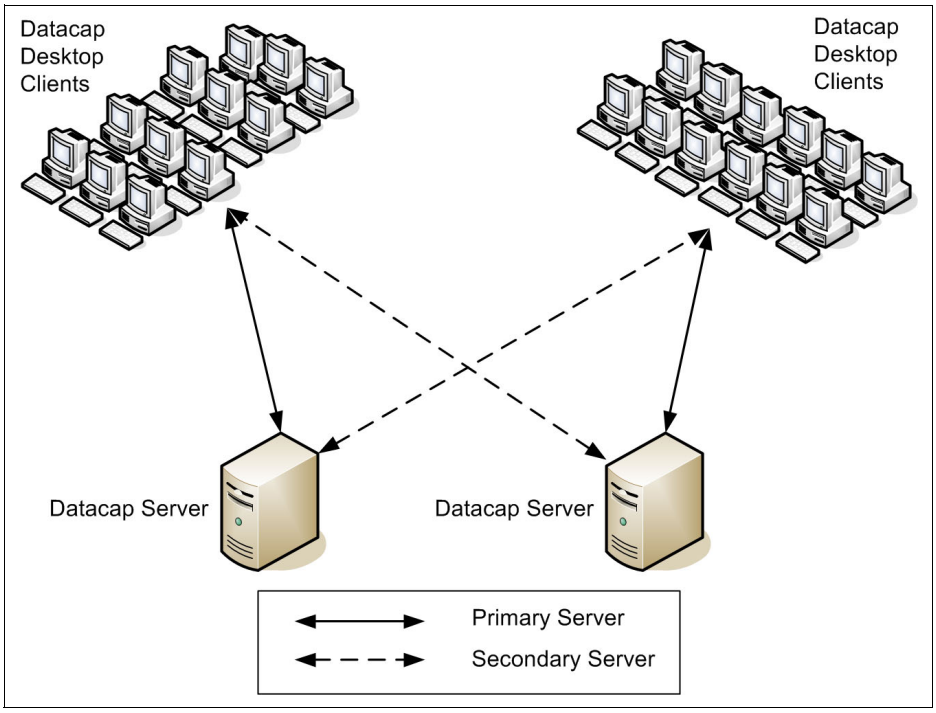


Figure 8-10 Datacap Desktops connecting to multiple Datacap servers without a load balancer

8.1.11 Load balancing of tasks

Load balancing of unattended or background tasks is done in a simple way. Each Datacap Rulerunner thread polls the Datacap server that is requesting a list of batches that are pending for any one of the tasks configured. If a batch is available, the first thread to request it gets the work.

Datacap processes batches according to their priority level (1 - 10, with 1 being the highest priority) and then by age (FIFO). Batch priority can be assigned manually or by rules at any point in the workflow. All Datacap Rulerunner Servers poll at regularly defined intervals.

Datacap Rulerunner Server has a facility to place a polling priority on each task. The task polling priority is separate from the priority assigned to the batch. The polling priority for a task defines the number of times that a thread polls for the task. For more information, see “Mixed queuing mode” on page 194.

8.1.12 Scaling databases

IBM Datacap includes Microsoft Access as its standard database format. The two main databases are the Datacap Engine database and the Datacap Administrator database. The Datacap Administrator database holds the user credentials for users of the system. The Datacap Engine database maintains the status of each batch in the system.

Datacap also makes use of another key database called the Fingerprint database. This database maintains a record and location of all the Fingerprint templates held for a specific Datacap application. Fingerprint templates are then loaded during batch processing, or held in memory by the Fingerprint Server for use in page identification and zone definition.

For production implementations, use only enterprise-level databases that are currently supported by Datacap. Reserve use of the Microsoft Access databases only for demonstration or simple testing environments.

Note that use of multiple threads or multiple batch creation tasks in the same application that uses a Access database might cause database integrity issues.

For the current list of databases supported by your specific Datacap version, see Datacap system requirements website:

http://www.ibm.com/support/docview.wss?uid=swg27043811

To provide optimum performance and scalability in a high throughput environment, databases should be located on a separate dedicated server primarily used for this purpose. For small production systems, the database can share a server with Datacap server or Datacap Rulerunner.

Describing scaling of enterprise-level databases is beyond the scope of this book.

8.1.13 Network share drive

When Datacap ingests and processes documents, it stores them in a shared file system. This file system must be read/write accessible across the network by all Datacap servers, Datacap Rulerunner Servers and Datacap Desktop Clients.

Because batches contain multiple image files, the network bandwidth required for moving these files can get high. Therefore, sufficient bandwidth must be available between the client and the network share drive. The file system must also be mounted on a system that is capable of fast input and output.

A common configuration is to have a Datacap server fail over with a storage area network (SAN) drive that is connected to both.

Small scale implementations suffice on a desktop or on a small server system to provide their file system. For larger scale systems, with high throughput requirements, we suggest using a fully dedicated high-speed disk subsystem. Batches of documents can be archived to back up media after they complete the workflow. You can use Datacap Maintenance Manager for archiving.

8.1.14 Scaling across locations

Scanning of files in remote locations and storing them in remote network shared drive can present performance issues if the network is relatively slow. Datacap provides capabilities to overcome some of these issues and we describe these in this section.

Separate Datacap instances

A simplified approach is to install separate instance of Datacap in the different locations. In this approach, each location has their own Datacap components installed locally. Figure 8-11 shows a possible distributed, Datacap Desktop implementation of IBM Datacap.

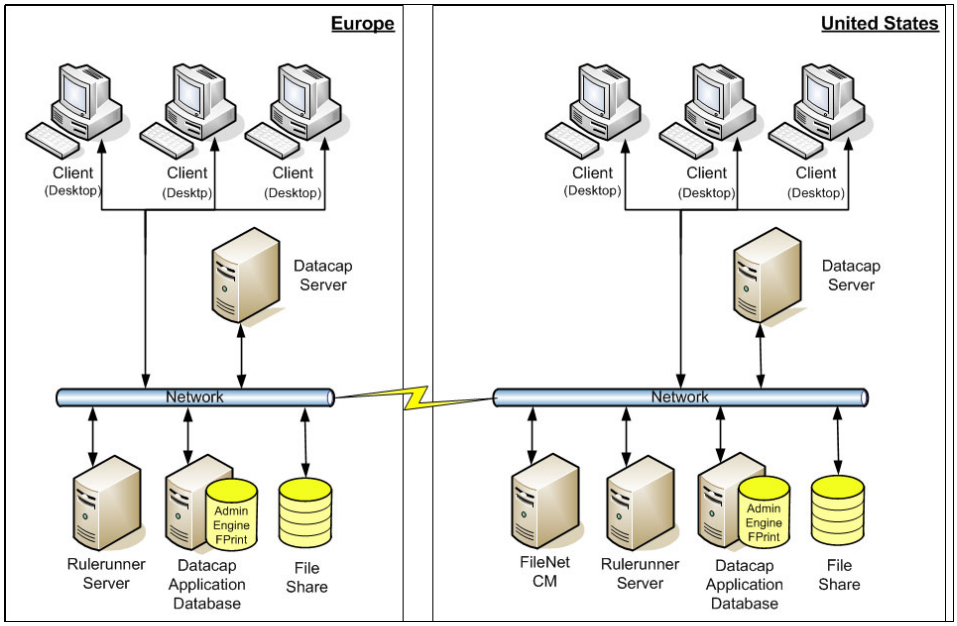


Figure 8-11 Distributed Datacap implementation

All the scanned files and batch data are held on a network share drive on the same local area network (LAN) as the users, along with localized Datacap server, Datacap Rulerunner Servers, and database servers. Only when images and data are ready for committal, they are sent over the wide area network (WAN) to the Enterprise Content Manager repository. Effectively you are running two separate Datacap instances.

Important: The application project files must be identical in both locations to ensure that both systems work as intended.

The separation of having two Datacap servers also allows each geography to operate independently of the other, offering a greater degree of resilience if a Datacap server fails.

Alternative configuration for your reference

DISCLAIMER: This alternative option is for your reference only. IBM has not tested nor provides support for this type of configuration. We provide this description strictly for your information.

Datacap allows use of wTM as described in 8.1.8, “Datacap wTM Server scaling and redundancy” on page 182. Its capability provides possible solutions to bridge applications together in both local and remote locations.

Through use of Datacap Web Services server and web service actions, Datacap offers the ability to interface from one application to another, either in the same or remote locations. For example, batches created in a central Input Director application can be initially processed and then routed, depending on the content, to another child application located elsewhere specifically designed for the type of document to process. This interface also permits the copying of any batch or page variables and other associated batch files to the child application. This delivers a complete audit trail back to the batch or point of origin in the Input Director application.

An example is available for download at the Datacap Technical Mastery forum on IBM developerWorks:

http://ibm.co/1f49YLY

Another alternative configuration to scale your system across locations is to run a centralized Datacap server, database server, and file share. However, network bandwidth issues might become apparent when trying to send large quantities of scanned and verification data over the network in remote or low network bandwidth locations.

To overcome this network bandwidth issue, consider using localized network file shares to store batch data. This approach removes the need to repeatedly move files across potentially low-bandwidth network connections. Instead only move them at the point of committal (if required). This way, you can use a centralized Datacap server to manage all batches and localize clients and Datacap Rulerunner Servers to process the data. Use separate localized `datacap.xml` files to implement this configuration.

8.2 Datacap Rulerunner

Datacap Rulerunner Servers run tasks that do not require human interaction such as optical character recognition (OCR), intelligent character recognition (ICR), optical mark recognition (OMR), bar code recognition, and export to repository. This is typically the most process intensive server and hardware should be optimized for use.

Datacap Rulerunner Server can be used in a single-thread or multi-thread configuration. Such configurations require attention to load balance, race condition, and more as explained in this section.

8.2.1 Single-threaded Datacap Rulerunner

Datacap Rulerunner Server can be configured to use a single thread on each Datacap Rulerunner Server, physical machine, or VMware. By using a single thread, processing of Datacap tasks is done in serial, with one process being run at any one time on each Datacap Rulerunner Server.

Figure 8-12 shows the process of a single thread within Datacap Rulerunner and how it interacts with the Datacap server, batch folder, Engine database, and Administrator database. This process is repeated each time a task is run by a Datacap Rulerunner thread.

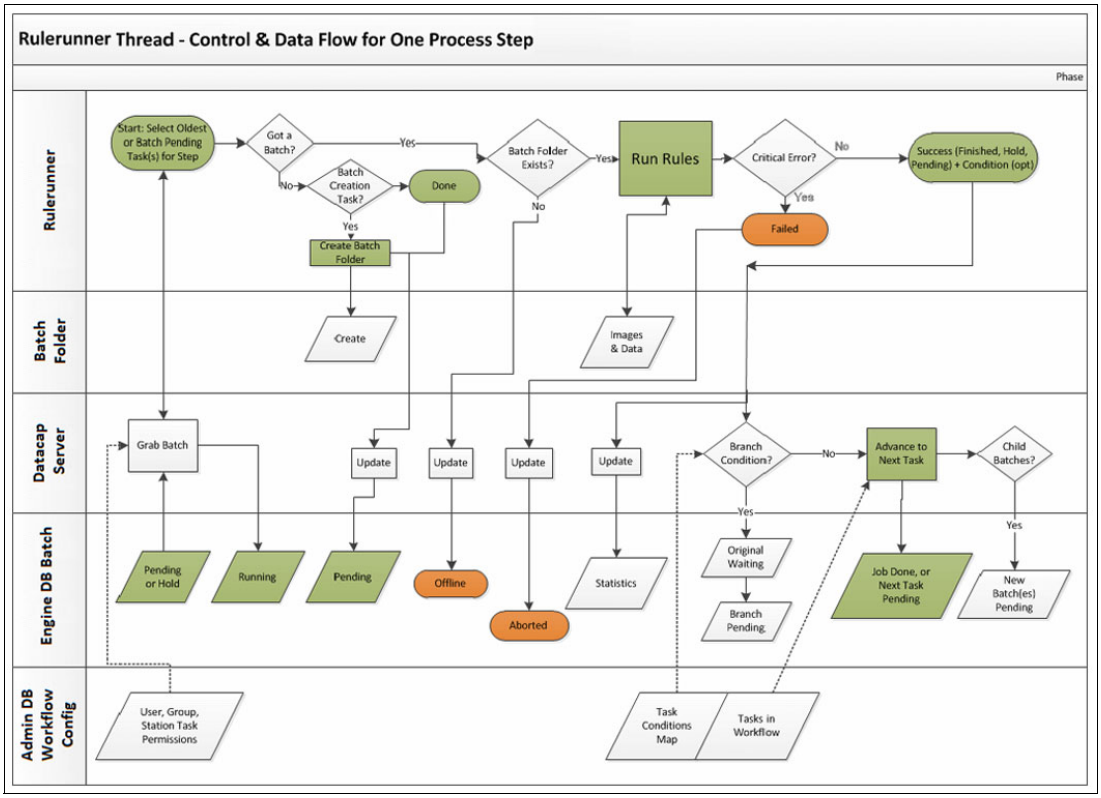


Figure 8-12 Datacap Rulerunner process

The disadvantage of using a single thread is that you do not use the full potential of the hardware that you have available. For example, if you have a quad-core processor capable of running 6 or more threads, you only gain approximately one-sixth of the potential throughput of the server.

8.2.2 Multi-threaded Datacap Rulerunner

Multi-thread licensing: At the time of publication, additional licensing is required on top of the Standard Datacap license for use of multithreading in Datacap Rulerunner. See your IBM sales representative for more information.

Modern day systems that have multiple processor cores allow support for multithreading. To scale up and use this additional processing power, use of multiple threads is advantageous for each physical machine or VMware.

There is no limit to the number of threads Datacap Rulerunner can manage. However, an optimal number needs to be determined depending on specific requirements, such as the type of task you are running. If you set the number of threads too high, the system might use up all resources, and performance will degrade.

The ratio of threads to processor cores can be a ratio of 1:1, 4:1, or greater. Processing speed and throughput vary depending on the task and the type and speed of the processor used. The number of threads can be calculated as an approximation, but to get a more accurate number, they must be determined experimentally to ensure optimum performance.

Typically one thread processes a task faster than when four threads process the same identical task on the same physical or VM instance. For example, a Datacap Rulerunner Server configured to run with only one thread might process a page in 4 seconds. If you now create four threads and process the same task on each one, it might take 5 seconds to process a page for each thread.

If you take this example and look at processing over a minute, in a single-thread mode, the system processes approximately 15 pages. In multi-thread mode, the system processes approximately 48 pages. This correlation is not a direct 1:1 relationship. In this example, 15 pages multiplied by 4 equals 60 (`15 X 4 = 60`), but only 48 pages are processed for the multi-thread mode with 4 threads. However, the overall throughput is still higher than in single-thread mode, and you have not increased the footprint of your server room.

Datacap Rulerunner is a 32-bit application. The function of rulerunner.exe is to manage the child RRProcessor.exe processes for each configured thread. Each thread managed by Rulerunner can have 2-3 GB of memory available.

When Datacap Rulerunner creates a RRProcessor.exe process, it monitors the process to detect if any issues occur. A timeout can be configured, after which it considers the process to be hung. Datacap Rulerunner can also be configured to stop its own service if any task stops. A background service detects if Datacap Rulerunner stops or hangs and then attempts to restart it. Logging can also be switched on to aid in troubleshooting any issues that might arise.

Figure 8-13 on page 193 shows a single-thread Datacap Rulerunner Server running tasks A, B, C, and D in order on a quad-core server, not using the additional processing capability of those cores. The order in which the tasks are run is repeated in a constant loop. (This is true if Sequential Queuing is selected. In Mixed Queuing, the highest priority, oldest batch for any of the selected tasks is selected.)

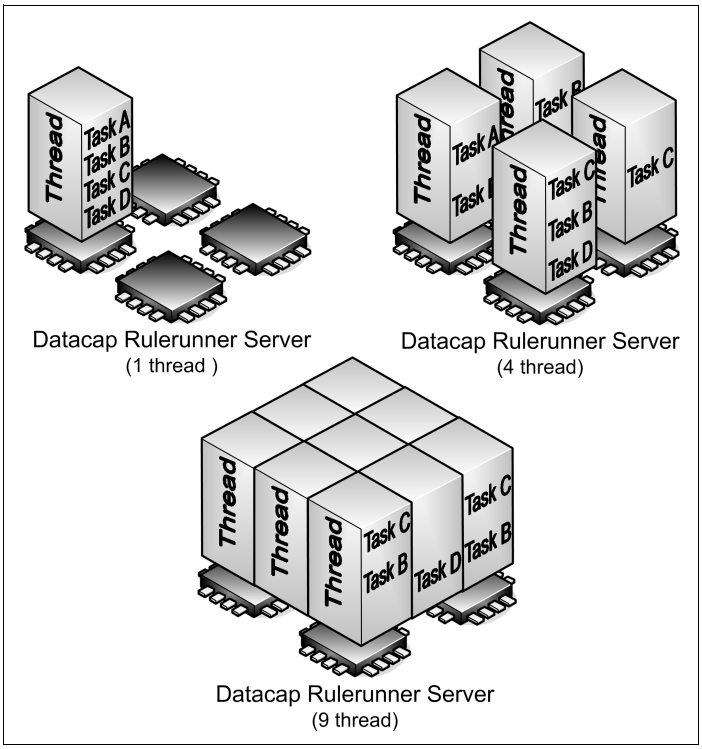


Figure 8-13 Datacap Rulerunner threads

The four-threaded configuration shows different tasks being run simultaneously. Some threads are dedicated to a single task, such as task C in the example. Some threads share the load by running different tasks, similar to the single threaded configuration. The multi-thread configuration uses the multiple cores that are available on the machine.

The nine-threaded configuration shows different tasks being run simultaneously. Some threads are dedicated to a single task. The configuration also shows that the number of threads is not restricted to the number of processor cores of the server. The number of threads must be optimized depending on the tasks that are being processed.

For a walkthrough of a basic Datacap Rulerunner configuration, see 8.3, “Configuring Datacap Rulerunner” on page 196. That same section explains how to achieve single threading and multithreading.

8.2.3 Sequential and mixed queuing

Datacap Rulerunner can be configured to run in both sequential queuing mode and mixed queuing mode. Choosing the correct option is important to scale your system effectively.

Sequential queuing mode

In sequential queuing mode, when all priorities are equal, the order in which the tasks are processed is defined by the order in which they are configured in Datacap Rulerunner Manager. The query that is sent to Datacap server consists of only one job for each task pair.

If tasks are given different priorities in Datacap Rulerunner Manager, Datacap Rulerunner performs the higher priority task on multiple batches before advancing to perform the next task. For example, Thread 1 has Task A with a priority of 8 and Task B with a priority of 2.

Priority is calculated based on the lowest common denominator. Therefore, the ratio of 8:2 is simplified to 4:1.

Because Task A is displayed in the thread before Task B, Task A goes first. The thread polls for work from the Datacap server and processes the highest priority, with the oldest available Task A batch available that is pending. It polls four times. Because Task B is displayed next, the thread now polls Datacap server for the highest priority, with the oldest available Task B batch pending. It polls one time.

Mixed queuing mode

In mixed queuing mode, the task priority that is set in Datacap Rulerunner is ignored. Datacap server returns the highest priority batches in FIFO order from *all* of the available job and task pairs in the thread. The first batch in this list is processed.

Mixed queuing mode is not appropriate if Batch Creation tasks, such as MVScan, are configured with any other tasks.

In mixed queuing mode, Datacap server always selects the next batch. The position of the tasks in the thread is unimportant.

For an example, see 8.3.2, “Configuring priorities and queuing within Datacap Rulerunner Server” on page 199.

8.2.4 Race conditions

When defining which tasks to run on a specific Datacap Rulerunner Server or thread, consider the fact that certain actions use the same resources. For example, a task such as VScan polls a defined directory for available files to consume and then creates a batch. If the same task is running in a separate thread or separate Datacap Rulerunner Server attempts to run this task, it might encounter issues. Files that are supposed to be consumed in serial order might be broken into separate batches, corrupting the scanning order. This issue is largely addressed in Datacap 9 by the use of VScan to ensure two or more batch creation tasks do not simultaneously grab the same files.

Similarly, a task might require access to a specific file. The first task places a lock on this file until it completes. Therefore, other tasks that also require this file must wait until the lock releases before they can continue. This process affects the performance of subsequent threads that are each waiting for the lock on the file to be released.

When using multiple servers to implement a solution, separate the tasks that might share local resources or that are not thread safe onto different physical machines or virtual machines where possible. This approach helps to avoid any conflicts that might result in an incorrect operation. Also consider how each task or action works, and then test accordingly to minimize any multiprocessing issues.

Important: Consider and test thread safety and multiprocessing when writing custom actions.

8.2.5 Running Datacap Rulerunner in virtualized environments

Datacap Rulerunner is supported for use on VMware. Compared to running it natively, running on VMware results in a drop in performance. Therefore, to achieve optimal performance, while running multiple processes, you might want to install Datacap Rulerunner onto a native Windows operating system without the virtual machine.

In certain virtual environments, use of CPU affinity (dedicated CPUs allocated to a virtual machine) has been seen to improve performance.

8.2.6 Fingerprint Service

Additional license: Depending on the type of license purchased, an additional license might be required for use of the Fingerprint Service. See your IBM Sales representative for more information.

As explained in previous chapters, IBM Datacap uses fingerprinting technology to identify documents as they enter the system. In large-scale IBM Datacap implementations, where large numbers of fingerprints (typically over 1000 fingerprints) are required for document identification, use of the Fingerprint Service is suggested.

The Fingerprint Service overcomes the time-consuming process of loading all the fingerprints of the application each time a batch is run. The solution involves reading all the fingerprints into the system memory cache of the Fingerprint Server when the first fingerprint match is requested. To further optimize this process, only the identification portion of the fingerprint is loaded into memory. The fingerprints remain in system memory for the life of the Fingerprint Service process.

When the Fingerprint Service is requested for the first time, it loads all `.cco` fingerprint files that are defined in the Fingerprint database. The greater the number is of Fingerprints to read into memory, the longer the service takes to start, which can be from a few seconds to a few minutes.

During the startup process, if another Datacap Rulerunner workstation tries to call the Fingerprint Service, the second request pauses until the Fingerprint Service completes loading all fingerprints.

If new fingerprints are created on demand by using the Click ‘N Key and Intellocate functions, the new fingerprint is loaded into the service. Similarly, if an image that is recognized returns a fingerprint match from the Fingerprint Service that no longer exists in the Fingerprint database, the image is removed from the Fingerprint Service memory. Then the image is re-queried for another fingerprint match. This method negates the potentially lengthy process of reloading the Fingerprint Service when additions or deletions occur.

To use the Fingerprint Service in a project, you use the `SetFingerprintWebServiceURL` and `SetApplicationID` actions in the Autodoc Global action library and point them to the server URL that is running the service. It is also specified in the FastDocUser Interface when creating a Form template application, as shown in Figure 8-14.

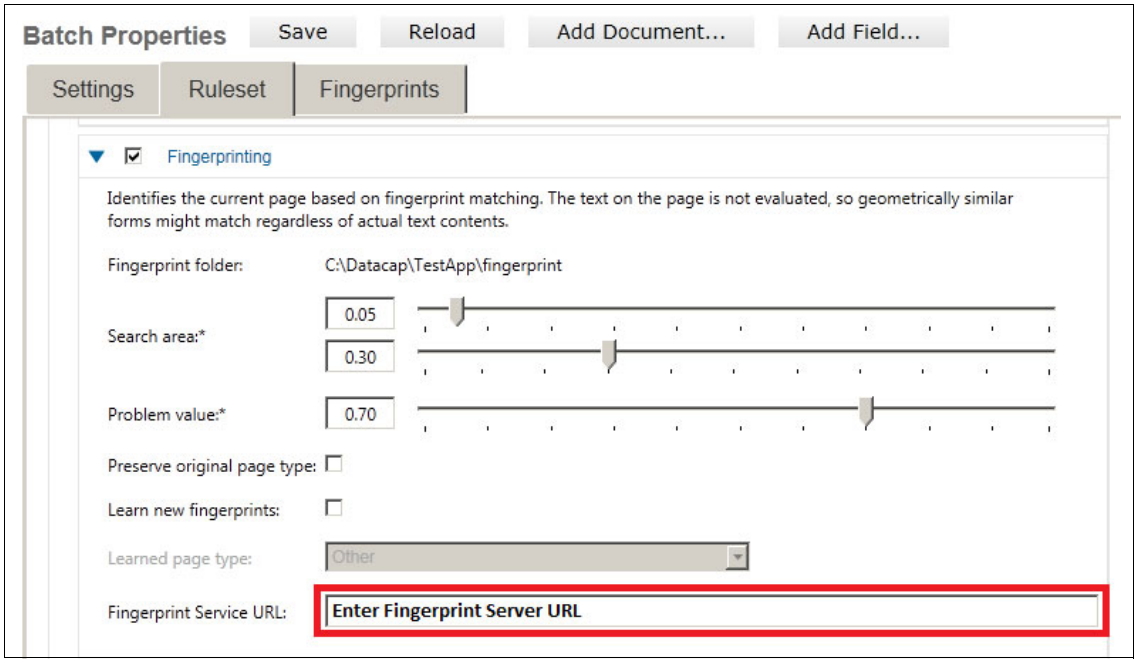


Figure 8-14 FastDoc Form template user interface showing FIngerprint Service URL configuration

A stand-alone application, Datacap Fingerprint Service Test Tool, is available to test the Fingerprint Service. This application enables you to add, search, and unload the fingerprints of a project if necessary. For more information about installation and testing, see “Installing and configuring the Datacap Fingerprint Service” in IBM Knowledge Center:

http://ibm.co/1Ph757F

8.3 Configuring Datacap Rulerunner

This section outlines a basic configuration of Datacap Rulerunner. It uses a fictional Auto_Claim Datacap project.

We make the assumption that you have installed one or more Datacap Rulerunner Servers in a test environment. For information about setting up a system with appropriate security and other considerations, see “Installing and configuring the Rulerunner Service” in the Datacap section of IBM Knowledge Center:

http://ibm.co/1Ph6XoF

The servers ECMDEMO1 and ALPHA are used in this example. ECMDEMO1 is the central server on which all Datacap components are installed. ALPHA is initially clean and has an additional Datacap Rulerunner and an additional Datacap server installed during the following process.

8.3.1 The Datacap.xml and <project>.app files

To configure the application to run in Datacap Rulerunner, use the Datacap Application Manager. This application provides a user interface so that you can change Datacap applications defined in the `datacap.xml` file:

```
c:\Datacap\datacap.xml
```

The `datacap.xml` file contains the centralized “Datacap Application Service” settings that are used by all Datacap components in your system. For Datacap Rulerunner to run, it requires the location of the central `datacap.xml` file by using a file path (the UNC path for separate server installations):

```
C:\datacap\datacap.xml (can be used for single-server installation)
```

or

```
\\ECMDEMO1\datacap\datacap.xml (should be used for multi-server installation)
```

Example 8-1 shows a possible configuration of the `datacap.xml` file pointing to various projects on different servers. For Datacap Rulerunner to process tasks for a given application, it must be able to gain access the `datacap.xml` file and to the file shares for the applications defined in this file. Ensure that the user that you use for the Datacap Rulerunner Service has relevant privileges to access these locations. You can configure this in the Windows Services console.

Example 8-1 The datacap.xml file

```
<datacap ver="8.0">
<app name="Auto_Claim" ref="\\ECMDEMO1\Datacap\Auto_Claim"/>
<app name="Flex" ref="\\DEMO2\Datacap\Flex"/>
<app name="1040ez" ref="\\SDEMO2\Datacap\1040ez"/>
</datacap>
```

To function, the Datacap components must know the location of the `datacap.xml` file. If you use a single-server installation, the location is already set at installation time to the `C:\datacap\datacap.xml` directory.

After the application is reached, Datacap Application Manager can query the `.app` file in the project folder as follows and as used in this example:

```
C:\Datacap\Auto_Claim\Auto_Claim.app
\\ECMDEMO1\Datacap\Auto_Claim\Auto_Claim.app
```

Figure 8-15 on page 198 shows the Service tab of Datacap Rulerunner Manager configured to point to the `Datacap.xml` file by specifying its full UNC-formatted path.

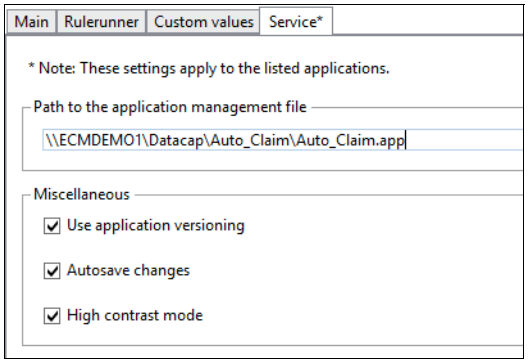


Figure 8-15 Datacap Rulerunner Manager Service tab pointing to datacap.xml

The `.app` file holds all the project paths, connection strings, and other settings that are used by applications to reference the batch folder. Although you can manually alter the `.app` file, use the Datacap Application Manager instead. Using Datacap Application Manager ensures that XML tags are not omitted and that connections strings for databases and other values are correctly encrypted.

Example 8-2 shows a sample of the `.app` file that is used for Auto_Claim project.

Example 8-2 The auto_claim.app file

```
<app name="Auto_Claim" ver="83" modder="Administrator.ECMDEMO1.ECM"
dt="07/08/11.826  10:08:02.826 " src_ver="53">
   <k name="tmservers">
      <k name="tms" ip="ECMDEMO1" port="2402" retry="3"/>
   </k>
   <k name="runtime" v="batches"/>
   <k name="tmengine"[secured]01671498c6aa12b8aaddb9ea348[/secured]"/>
   <k name="tmadmin" cs="[secured]8c6aa12b8aagFiZ37g5pr5E[/secured]"/>
   <k name="dco_Auto_Claim">
      <k name="setupdco" v="Auto_Claim.xml"/>
      <k name="rules" v="rules"/>
      <k name="imagefix" v="imagefix.ini"/>
      <k name="UseFPXML" v="True"/>
      <k name="fingerprintconn" cs="[secured]6OzG8c6aa12Y[/secured]"/>
      <k name="lookupdb" cs="[secured]Jj8c6aa12b8YVCUTBSe[/secured]"/>
      <k name="vscanimagedir" v="\\ecmdemo1\Datacap\Auto_Claim\image"/>
      <k name="exportdb" cs="[secured]8c6aa12b8aafehwjwjs[/secured]"/>
   </k>
   <k name="fingerprint" v="fingerprint"/>
   <k name="export" v="export"/>
   <k name="tasks">
      <k name="VScan" profile="VScan"/>
      <k name="PageID" profile="PageID"/>
      <k name="Rulerunner" profile="Rulerunner"/>
      <k name="Export" profile="Export"/>
   </k>
</app>
```

By use of the `datacap.xml` and `<application>.app` files, the location and settings of all projects can be obtained. Use of these files and smart parameters to remove all hardcoded paths and connection strings simplifies promotion from development to production.

Figure 8-16 shows the relationship between the `datacap.xml` file, the `.app` files for Auto_Claim and 1040ez, and the subsequent Datacap servers, file servers, and database servers that are used.

The `datacap.xml` file holds the list of applications and where they reside. The program accessing the `datacap.xml` file can use this location. Then it can open the `.app` file to extract the information it needs about the application, such as a database connection string, Datacap servers to use, batch directory location, and so on.

Figure 8-16 is an example of the component layout.

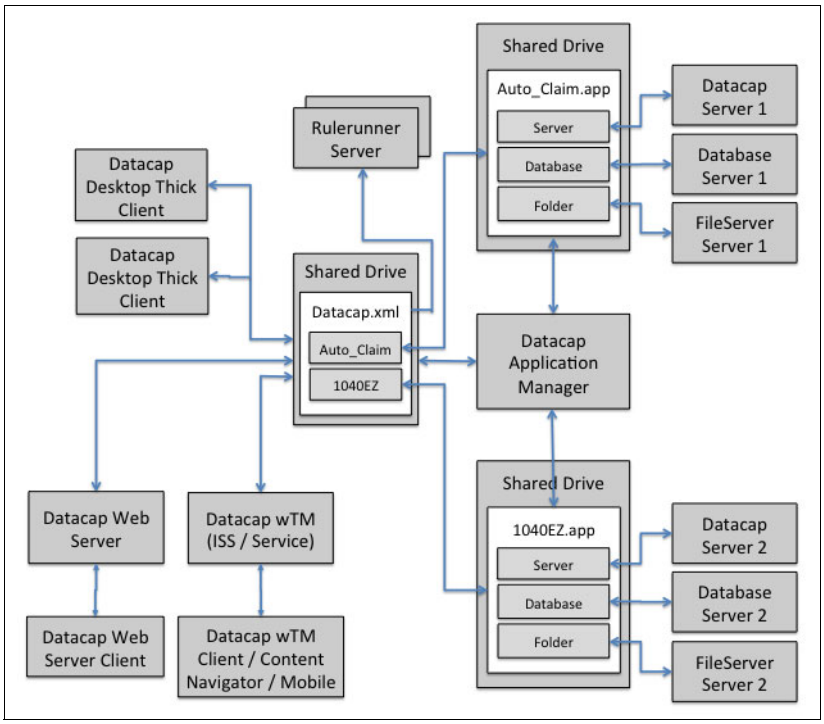


Figure 8-16 Datacap.xml file and .app file relationships

8.3.2 Configuring priorities and queuing within Datacap Rulerunner Server

As mentioned previously in this chapter, to scale your system accordingly, you can configure priorities against specific tasks and jobs. You can configure priorities and queuing in the Datacap Rulerunner Manager as shown in Figure 8-17. This is found by adding and configuring a thread in the Workflow:Job:Task tab of Datacap Rulerunner Manager by clicking a Task or Job.

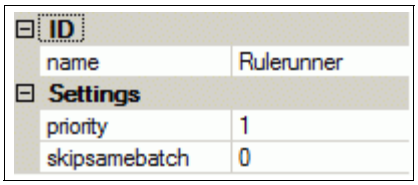


Figure 8-17 Configuring priority and skipsamebatch

After you added and saved a thread configuration to the `rulerunner.xml` file (typically saved in `C:\Datacap\taskmaster\Rulerunner.xml` but changeable in the Settings tab), you can look at the .xml configuration. You can see more detail about the parameters that are available for each thread, application, server, database, job, and task (see Example 8-3).

Use this file only as a reference. Do not change any of the values in the file. Instead, use the Datacap Rulerunner Manager to change the values.

Example 8-3 Example Rulerunner.xml file

```
<thread0 enabled="1">
<app name="Auto_Claim" priority="1">
      <server name="local" priority="1">
        <dbs admin="tmadmin" engine="tmengine" priority="1" >
          <job name="Main Job" priority="1">
            <task name="Batch Profiler" skipsamebatch="0" priority="1" />
            <task name="VScan" skipsamebatch="0" priority="3" />
          </job>
        </dbs>
      </server>
    </app>
    <app name="1040ez” priority="2">
      <server name="local" priority="1">
        <dbs admin="tmadmin" engine="tmengine" priority="1" >
          <job name="Demo" priority="1">
            <task name="VScan" skipsamebatch="0" priority="1" />
          </job>
        </dbs>
      </server>
    </app>
<thread0>
```

Look at the following important parameters in the XML file:

SkipSameBatch This is used when a task might set the batch back to a pending state. If you use VScan configured to remove images after creating the batch, when VScan runs again, and no images are available, the batch is reset to pending. To prevent running the batch repeatedly, this option introduces a delay. Therefore, if the Batch ID is the same next time it runs VScan, do not do anything for X seconds, effectively stopping a repeated loop until images are available.

Priority This is used to determine the ratio that the current thread needs to spend processing the current node among nodes of the same level. The priority is only applicable when more than one node exists at the same level. For example, more than one application (app) exists under one thread or multiple tasks (task) under one job.

The following queuing modes determine how the priority value is used. You can configure the mode in the Settings tab of Datacap Rulerunner Manager:

Mixed Priorities that are below database dbs level are ignored by Datacap Rulerunner. Priorities below the dbs level are determined by the Datacap server. Use mixed queuing only if you have a specific need.

Sequential The combined value of priorities of all levels is used to determine how often each particular job and task pair must be run.

The priorities are used to calculate a ratio. All the priorities are divided on the smallest value and rounded. For example, if we set priorities of 5 and 2, they are divided by a ratio of 3:1.

Thread-time distribution is based on the highest level node first. For example, multiple applications are specified, and the ratio is 1:2. If any priorities are set on the task level, the real ratio is combined with the higher level.

Example 8-3 on page 200 has two applications. thread0 queries batches from the 1040ez application twice more than it queries batches from the Auto_Claim application. The Auto_Claim application also has multiple tasks specified. These tasks depend on the queuing mode used:

- In mixed mode, it queries the Datacap server for the batch with highest priority among pending batches for Main Job/Batch Profiler and Main Job/VScan job and task pairs.
- In sequential mode, each time the thread queries the Datacap server, it first tries grabbing a batch for VScan for three times. Only on the fourth attempt, the Batch Profiler batch is grabbed.

In sequential mode, if you have an unlimited number of batches for all job and tasks pairs from all applications pending, it does the following processing:

1. Runs 1040ez VScan two times
2. Runs Auto_Claim VScan one time
3. Runs 1040ez VScan two times
4. Runs Auto_Claim VScan one time
5. Runs 1040ez VScan two times
6. Runs Auto_Claim VScan one time
7. Runs 1040ez VScan two times
8. Runs Auto_Claim Batch Profiler one time

If no batches are pending for any job and task pair, this job and task pair are queued next time according to the following formula:

```
s = 2^n;
```

where:

s Delay in seconds
n Amount of unsuccessful attempts to grab the batch

The maximum wait time will not exceed 64 seconds.

The sample set of values might look like the following process, where each step is an unsuccessful attempt to grab a batch:

1. Wait for 2 seconds
2. Wait for 4 seconds
3. Wait for 8 seconds
4. Wait for 16 seconds
5. Wait for 32 seconds
6. Wait for 64 seconds
7. Wait for 64 seconds
8. Successfully grabbed batch
9. Wait for 2 seconds
10. Wait for 4 seconds

If all job and task pairs have no pending batches, the thread goes to sleep for the time interval specified in the registry (default is 10 seconds). Try Fingerprint Service returning fingerprints added by an application.

8.4 Backing up and restoring

Ensure that backups are made of both the production and development systems so that you have a recovery point to revert to if a failure occurs.

Datacap 9 has a newly improved Application and Database Copy utility. This utility provides ease of copying between environments.

8.4.1 Backing up and restoring Datacap Rulerunner machines

Because Datacap Rulerunner Servers are stateless (process data and push it back to the batch directory and inform the Datacap server of its completion), no data is held on them. Therefore, no data is lost if the server fails. The only information that can potentially be lost is data that is in process at batch execution time.

As a result, backup is made much simpler. The Datacap Rulerunner Server must be brought down gracefully when no tasks are in process. After the server is brought down, a standard mirror of the whole server can be carried out.

If a failure occurs, the server can be restored to the point at which the mirror is made. Because no processes are running at the time of creation, the system can revert to a clean Datacap Rulerunner Server status. To successfully revert to a clean Datacap Rulerunner Server status, downtime for Datacap Rulerunner is essential.

After an initial backup is made, subsequent backups are needed only when changes or updates are installed on the Datacap Rulerunner Server such as fix packs.

8.4.2 Backing up and restoring the Datacap server

The Datacap server is the central Windows service that provides user authentication, workflow, and queuing (and file services for Datacap Web and wTM). It stores user details and the status of batches in two databases, which are the Administration and Engine databases.

Therefore, you must back up both databases to return to a preferred status. Backing up databases is well documented in many other publications and websites. The database backup procedure is beyond the scope of this IBM Redbooks publication and, therefore, is not addressed here. Consult your database vendor documentation for further information.

In a critical environment, use a database mirror so that you always have two copies of the database.

Backing up of the Datacap server ensures that you have a snapshot to restore to. You must close the Datacap server gracefully, ensuring that no tasks are currently in progress.

To carry out the shutdown, follow this sequence:

1. Where possible, close the connected clients.
2. Close any services.
3. Shut down the Datacap server.

After the system shuts down, you can carry out a standard mirror of the whole server.

If a failure occurs, the server can be restored to the point at which the mirror is made. Because no processes are running at the time of mirror creation, the system reverts to a clean Datacap Service status. When reverting to a mirror snapshot of the Engine database or the file share that contains document batches, any work processed between the time the mirror was created and the system failure is lost. To successfully back up the system, downtime for the system is essential.

The frequency of backup depends on the specific needs of the client.

8.4.3 Backing up the database server

Datacap uses Microsoft Access as its standard shipped database format. It uses two main databases, the Engine database and the Administrator databases. A Fingerprint database is also available.

Backing up and restoring databases is documented and beyond the scope of this book. Consult with your database vendor documentation for details.

8.4.4 Backing up and restoring the Fingerprint server

The Fingerprint Service, similar to Datacap Rulerunner, is stateless. It runs as a service and loads fingerprint details in the Fingerprint database. As a result, backing up is simpler because no fingerprints are stored on its file system.

Bring down the Fingerprint Service gracefully when no tasks are in process. After you bring it down, a standard mirror of the whole server is sufficient.

If a failure occurs, you can restore the server to the point at which the mirror is made. Because no fingerprints are loaded into memory at the time of creation, the system can revert to a clean Fingerprint Server Service. To successfully restore the system, downtime for the system is essential.

The frequency of backup depends on the specific needs of the client.

8.4.5 Backing up and restoring the IIS web server

The IIS web server serves pages to the web for users to scan, verify, and administer the system. When images are scanned, they are held on the IIS web server before they are uploaded. Apart from this process, the server is stateless. It holds no other permanent data.

Bring down the IIS web server gracefully when no tasks are in process. After the server is down, a standard mirror of the whole server is sufficient.

If a failure occurs, you can restore the server to the point at which the mirror is made. The system can revert to a clean IIS web server status. To successfully restore the system, downtime for the system is essential.

The frequency of backups depends on the specific needs of the client.

8.4.6 Backing up the file share

The file share holds all batch data and the Datacap application project files (if you store them there). Continuously mirroring this drive ensures that you have a fully current version of the batches. If it is paired with database mirroring of the Engine database, it also ensures the status of that batch.

9

Export and integration

This chapter showcases IBM Datacap abilities to export captured data and documents to a repository. We focus on two concepts:

- Data and document formatting to prepare captured content for export
- Ways to integrate and export content to external systems

9.1 Preparing content for export

Data and documents (content) must adhere to the specifications of the export repository. Data repository definitions might vary from database definitions. Documents might need conversion to specific file formats. Content might also need to be sent to separate repositories, as shown in Figure 9-1.

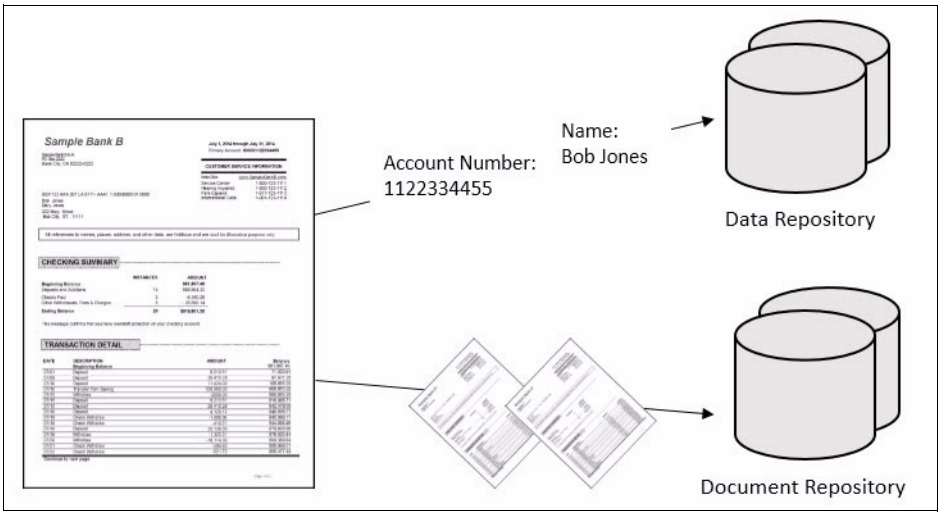


Figure 9-1 Data and documents exported to different repositories

Datacap contains multiple methods to format data. It provides the following native abilities:

- Export data to a flat text file
- Export data to an XML file
- Export data into a database

We examine each of these methods in this chapter.

9.2 Formatting data for export

Data is collected during the capture process. When exported, the data must be in a format the repository requires. In this section, we take two scenarios and show how a Datacap application can address each.

The first scenario reformats a captured date value. We take a date formatted as dd.mm.yy and reformat it as `yyyy.mm.dd`. “Export the Date value to a flat text file” on page 208 describes how to accomplish this by using the value shown in Table 9-1.

Table 9-1 Date field and value before reformatting

Field name	Captured value
CaptureDate	01/01/15

The next scenario takes multiple captured values and assembles them to create a new one. We use the reformatted date from the first scenario. We then combine it with other values to create a value named `CreatedBy` in this format:

`CreatedBy: LastInitial, FirstName on FormattedDate`

“Export CreatedBy value to a flat text file” on page 209 explains how to accomplish this by using the values in Table 9-2.

Table 9-2 Fields and values before assembly

Field name	Captured value
FormattedDate	2015/01/01
FirstName	Sparky
LastInitial	B

Datacap offers flexibility exporting data to a flat text file. The Datacap Export library contains over thirty actions. Figure 9-2 shows a partial list of actions.

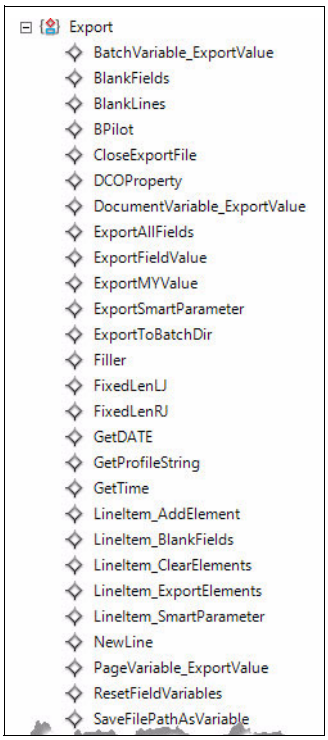


Figure 9-2 Partial list form Export library actions

Export the Date value to a flat text file

To reformat and export the `CaptureDate` value, we first set the export file attributes. We then reformat the date value and export it to the flat text file. The `TXTExportDate` ruleset in Figure 9-3 shows one way to accomplish this.

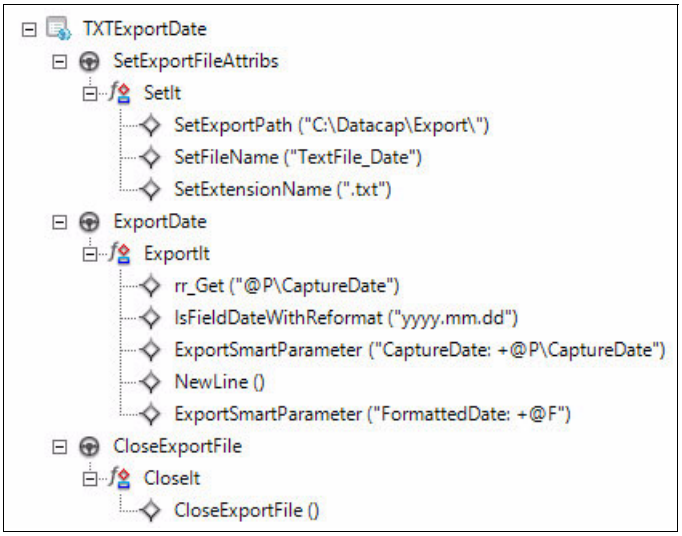


Figure 9-3 Reformat and export date ruleset

The functions and actions shown in Figure 9-3 complete the following tasks:

- The `SetIt` function within the `SetExportFileAttribs` rule performs these actions to set the export file attributes:
  - Sets the export file location (`SetExportPath`)
  - Sets the export file name (`SetFileName`)
  - Sets the export file extension (`SetExtensionName`)

  This function is normally associated with the document element of the document hierarchy (DCO). This creates a separate file for each document. Alternatively, if it is associated with the batch element, a single file that contains all of the exported data is created.
- The `ExportIt` function within the `ExportDate` rule reformats the CaptureDate and outputs data to the export file, where it completes these tasks:
  - Places the `CaptureDate` field value in the `FormattedDate` field (`rr_Get`)
  - Reformats the `FormattedDate` field value (`IsFieldDateWithReformat`)
  - Writes the CaptureDate field value to the export file (`ExportSmartParameter`)
  - Writers a hard return to the export file (`new_line`)
  - Writes the FormattedDate field value to the export file (`ExportSmartParameter`)

  This function is associated with the `FormattedDate` field because the `rr_Get` action populates the value of the field that it is associated with.

Figure 9-4 shows the output after reformatting the date value with the ruleset that was shown in Figure 9-3.

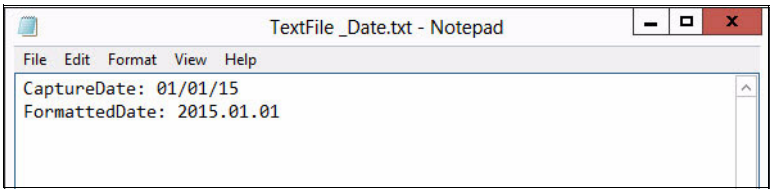


Figure 9-4 Flat text file created with ExportDate rule

Export *CreatedBy* value to a flat text file

In this section, we construct a value named CreatedBy. We use only smart parameters in the ExportCreatedBy function to export the data. Figure 9-5 shows the TXTExportCreatedBy ruleset to accomplish this.

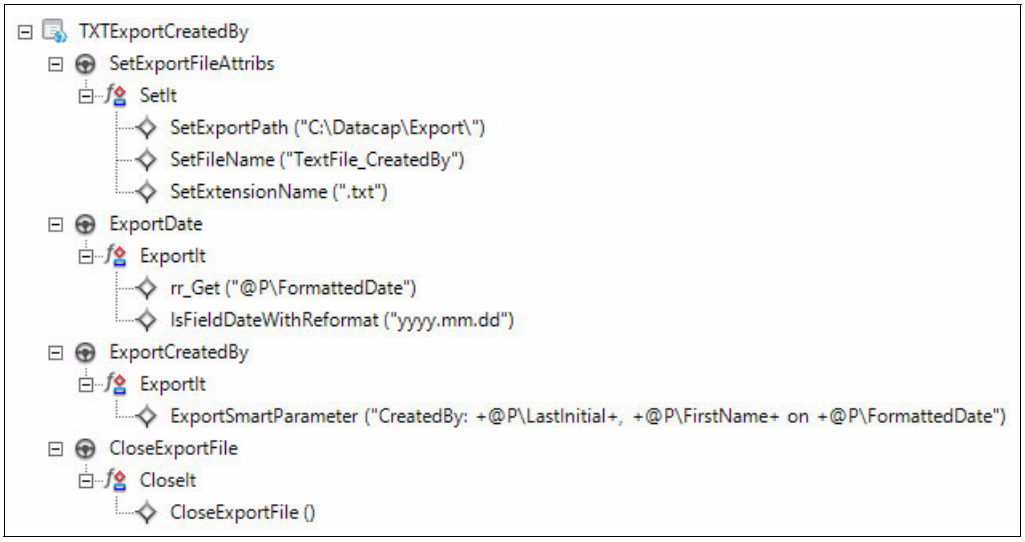


Figure 9-5 Build and export CreatedBy value ruleset

> **Tip:** Smart parameters reduce or eliminate hardcoding of action parameters. In Figure 9-3 on page 208, all text file attributes are hardcoded. Also, the value names are hardcoded in the export actions but the values are pulled directly from the DCO field.

The following list describes what each function and action does, as shown in Figure 9-5:

- The `SetIt` function within the `SetExportFileAttribs` rule uses the following actions to set the export file attributes:
  - Sets the export file location (`SetExportPath`)
  - Sets the export file name (`SetFileName`)
  - Sets the export file extension (`SetExtensionName`)

  As in the previous example, this function would normally be associated with the document element of the DCO. This creates a separate file for each document. Alternatively, if associated with the batch element, a single file containing all the exported data is created.
- The `ExportIt` function within the `ExportCreatedBy` rule uses a single action to write the `CreatedBy` value to the flat text file. This function constructs and exports the `CreatedBy` value (`ExportSmartParameter`).

  Notice the parameter values of the action. Beside the label and value delimiter, the value is constructed using smart parameters.

Figure 9-6 shows the output after running the ruleset shown in Figure 9-5 on page 209.

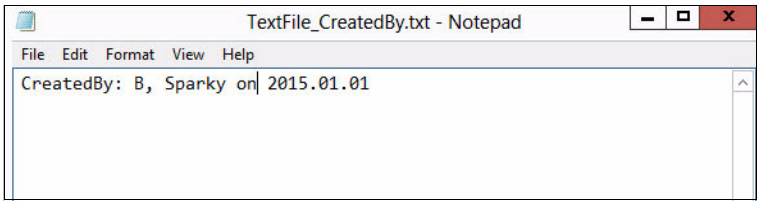


Figure 9-6 Flat text file created from ExportCreatedBy rule

9.2.1 Export data to an XML file

The XML Export library is native to Datacap. Data within any element of the DCO can be output to an XML file. Figure 9-7 shows the actions available for generating an XML file.

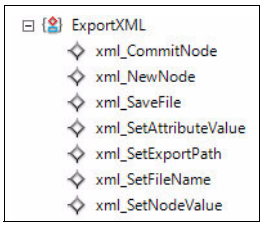


Figure 9-7 Datacap XML export actions library

Export the Date value to an XML file

Here we demonstrate exporting a series of captured values to an XML file. Table 9-3 shows the values that we export in this scenario.

Table 9-3 Vales for exporting to an XML file

Field name	Captured value
CaptureDate	01/01/15
FormattedDate	2015.01.01
Author_I	Nancy B
Author_II	John B
Author_III	Lisa B
Author_IV	Katherine B
Author_V	Megan B
Author_VI	Alyson B

Figure 9-8 shows the Datacap ruleset for building and exporting values to an XML file.

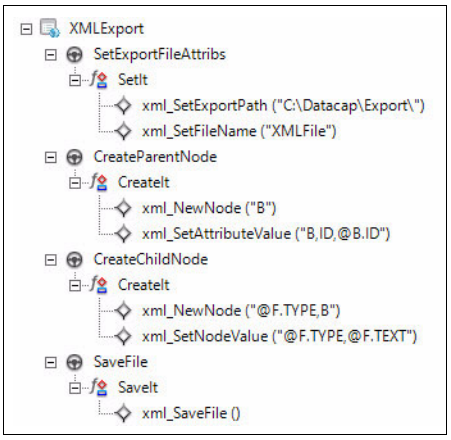


Figure 9-8 Build and export values to an XML file ruleset

The following list describes what each function and action does, as shown in Figure 9-8:

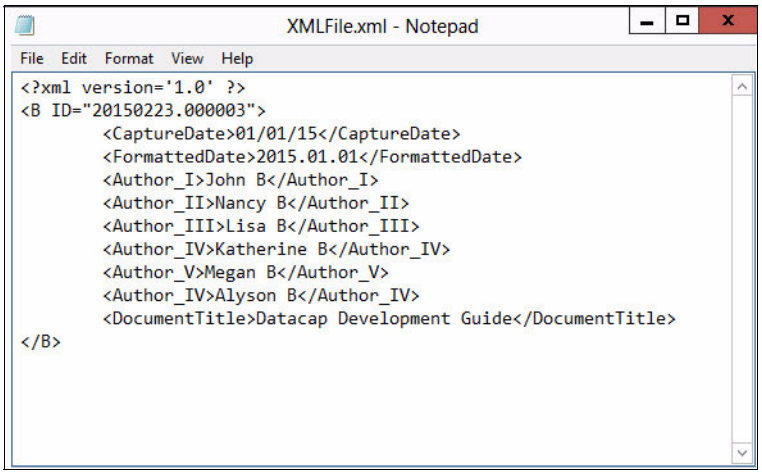
- The `SetIt` function within the `SetExportFileAttribs` rule uses the following actions to set the export file attributes:
  - Sets the export file location (`xml_SetExportPath`)
  - Sets the export file name (`xml_SetFileName`)

  This type of function is commonly associated with the document level to ensure a single XML file is created for each document in the DCO.
- The `CreateIt` function within the `CreateParentNode` rule uses the following actions to create the parent node:
  - Creates a node named “D” (`xml_NewNode`)
  - Assigns attributes to the specific node (`xml_SetAttributeValue`)

  This type of function is commonly associated with the document level when the node being exported is a container node for document values.
- The `CreateIt` function in the `CreateChildNode` rule uses the following actions to create a child node within the D node:
  - Creates a node named the same as the DCO field name (`xml_NewNode`)
  - Assigns attributes to the specific node (`xml_SetAttributeValue`)

  This type of function is commonly associated with the field level when the node being exported is a value that is contained in a field.
- The `SaveIt` function writes the XML structure out to the specific file.

Figure 9-9 shows the result.




```
<?xml version='1.0' ?>
<B ID="20150223.000003">
        <CaptureDate>01/01/15</CaptureDate>
        <FormattedDate>2015.01.01</FormattedDate>
        <Author_I>John B</Author_I>
        <Author_II>Nancy B</Author_II>
        <Author_III>Lisa B</Author_III>
        <Author_IV>Katherine B</Author_IV>
        <Author_V>Megan B</Author_V>
        <Author_IV>Alyson B</Author_IV>
        <DocumentTitle>Datacap Development Guide</DocumentTitle>
</B>
```

Figure 9-9 Output of values to XML file

Note: You can create nested parent-child relationships. This is often seen in the invoice processing application, as shown in Figure 9-10.

```
<F id="Taxes">
        <V n="TYPE">Taxes</V>
        <V n="Position">0,0,0,0</V>
        <V n="STATUS">0</V>
        <V n="PreVerify Val"></V>
        <V n="PreVerify Pos">0,0,0,0</V>
        <V n="label">Taxes</V>
        <F id="TaxLineitem1">
                <V n="TYPE">TaxLineitem</V>
                <V n="STATUS">0</V>
                <V n="Position">0,0,0,0</V>
                <V n="PreVerify Val"></V>
                <V n="PreVerify Pos">0,0,0,0</V>
                <V n="label">TaxLineitem</V>
                <F id="Tax_Type">
                        <V n="TYPE">Tax_Type</V>
                        <V n="STATUS">0</V>
                        <V n="Position">0,0,0,0</V>
                        <V n="PreVerify Val">Sales</V>
                        <V n="PreVerify Pos">0,0,0,0</V>
                        <V n="label">Tax_Type</V>
                        <C cn="10" cr="0,0,0,0">83</C>
                        <C cn="10" cr="0,0,0,0">97</C>
                        <C cn="10" cr="0,0,0,0">108</C>
                        <C cn="10" cr="0,0,0,0">101</C>
                        <C cn="10" cr="0,0,0,0">115</C>
                </F>
                <F id="Tax_Value">
                        <V n="TYPE">Tax_Value</V>
                        <V n="STATUS">0</V>
                        <V n="Position">0,0,0,0</V>
```

Figure 9-10 Sample PageID XML file from APT application

Export data to a database

In conjunction with or independent of outputting data to a flat text or XML file, Datacap supports sending data to a database. This section shows in detail the procedure used to export captured data to a database.

Figure 9-11 shows the Datacap ExportDB actions library.

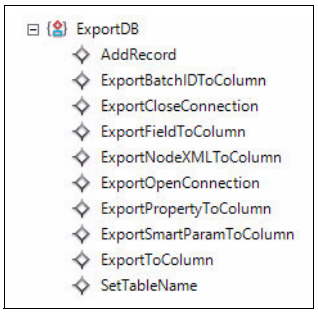


Figure 9-11 Datacap database actions library

As with the logic to create flat text and XML files, the `DBExport` ruleset (Figure 9-12) is similar.

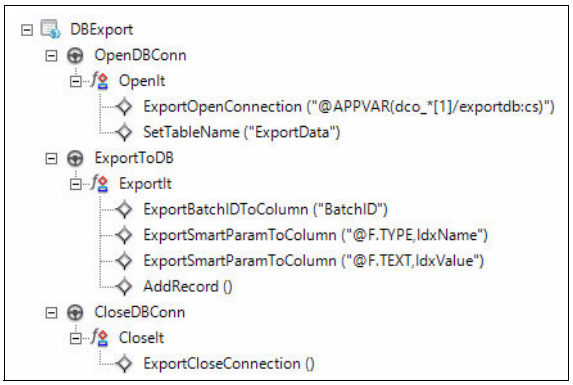


Figure 9-12 DBExport ruleset

The following describes what each function and action does, as shown in Figure 9-12:

- The `OpenIt` function within the `OpenDBConn` rule uses the following actions to set open a database connection and set the target table:
  - Opens a connection to the export database (`ExportOpenConnection`)
  - Sets the table data is exported to (`SetTableName`)

  The parameter for the `ExportOpenConnection` action is a reference to a variable stored in the IBM Datacap Application Manager (Figure 9-13).

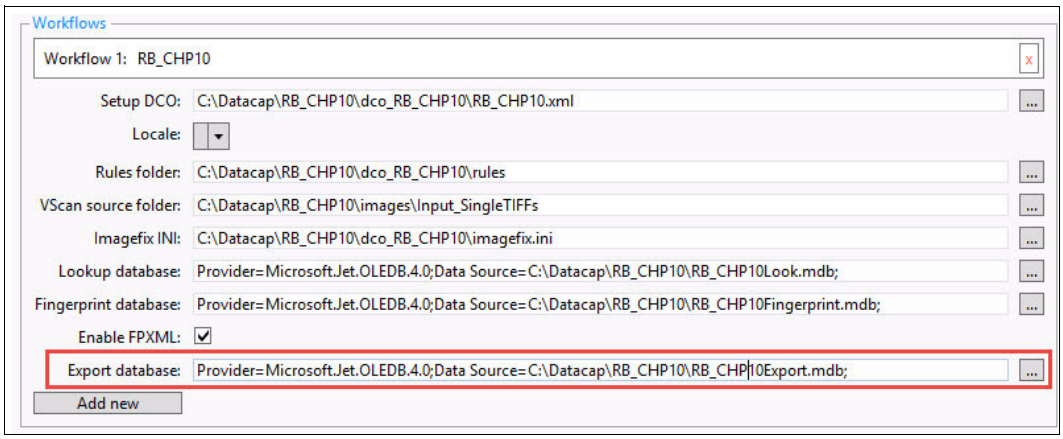


Figure 9-13 Export database connection value in the Datacap Application Manager

This type of function is commonly associated at the batch level in the DCO to output the batch data to a single database. Associating the SetTableName action to different elements in the DCO enables you to export data to multiple tables.

- The `ExportIt` function within the `ExportToDB` rule uses the following actions to export data to specified database and table:
  - Specifies the Datacap Batch ID be export *(`ExportBatchIDToColumn`)*
  - Specifies the field name be export (`ExportSmartParamToColumn`*)*
  - Specifies the field value be export (*E*`xportSmartParamToColumn`*)*
  - Writes the data to the table (`AddRecord`*)*

  Note: The database export actions build a record that is stored in memory until you commit them to the database using the *AddRecord* action.

  This type of function is commonly associated at the field level in the DCO.

- The `CloseIt` function within the `CloseDBConn` rule uses the following action to close the connection to the database:
  - Closes the opened DB connection *(`ExportCloseConnection`)*

  This type of function is commonly associated with the Close section at the batch level of the DCO.

Figure 9-14 shows a database table populated with captured values.

Tables

ExportData

ID	BatchID	IdxName	IdxValue
17	20150223.000014	CaptureDate	01/01/15
18	20150223.000014	FormattedDate	2015.01.01
19	20150223.000014	Author_I	John B
20	20150223.000014	Author_II	Nancy B
21	20150223.000014	Author_III	Lisa B
22	20150223.000014	Author_IV	Katherine B
23	20150223.000014	Author_V	Megan B
24	20150223.000014	Author_VI	Alyson B
25	20150223.000014	DocumentTitle	Datacap Development Guide

Figure 9-14 Database table populated with captured values

9.2.2 Formatting documents for export

The document being processed through the capture process is also, like data, expected to be in a format a repository can consume. To showcase this ability, we export the bank statement as a searchable PDF, JPEG, and multi-page TIFF file, as Figure 9-15 on page 215 shows.

Sample Bank B

Sample Bank B N.A.
PO Box 2222
Bank City, CA 02222-0222

July 1, 2014 through July 31, 2014
Primary Account: 000001122334455

CUSTOMER SERVICE INFORMATION

WebSite:	www.SampleBankB.com
Service Center:	1-800-123-1111
Hearing Impaired:	1-800-123-1112
Para Espanol:	1-877-123-1113
International Calls:	1-001-123-1114

0001122 AAA 001 LA 0111- AAAT 1 00000000 01 0000
Bob Jones
Mary Jones
222 Mary Street
Bob City, ST, 11111

All references to names, places, address, and other data, are fictitious and are used for illustration purposes only

CHECKING SUMMARY

	INSTANCES	AMOUNT
Beginning Balance		**$61,807.40**
Deposits and Additions	14	888,904.32
Checks Paid	5	-6,345.28
Other Withdrawals, Fees & Charges	8	- 25,565.14
Ending Balance	**20**	**$918,801.30**

This message confirms that you have overdraft protection on your checking account.

TRANSACTION DETAIL

DATE	DESCRIPTION	AMOUNT	Balance
	Beginning Balance		$61,807.40
07/01	Deposit	9,213.51	71,020.91
07/09	Deposit	26,410.29	97,431.20
07/10	Deposit	11,424.00	108,855.20
07/10	Transfer from Saving	700,000.00	808,855.20
07/13	Withdraw	-2000.00	806,855.20
07/15	Deposit	9,213.51	816,068.71
07/15	Deposit	26,410.29	842,479.00
07/18	Deposit	4,120.17	846,599.17
07/18	Check Withdraw	-1,500.00	845,099.17
07/18	Check Withdraw	-410.21	844,688.96
07/19	Deposit	35,138.00	879,826.96
07/19	Withdraw	-1,323.21	878,503.84
07/20	Withdraw	-19,114.00	859,389.84
07/21	Check Withdraw	-390.63	858,999.21
07/22	Check Withdraw	-521.72	858,477.49

Continue to next page

Figure 9-15 Sample bank statement

Export document as a searchable PDF

Exporting documents in various formats is routine. PDF format is arguably the most common. Datacap supports exporting documents in the following PDF formats, as of version 9.0:

- PDF/A
- PDF/A-1a
- PDF/A-1b
- Searchable
- Non-searchable

Data is captured from documents and regularly used to populate index values associated with documents in a repository. Indexes provide a way to search and retrieve the documents. There might be situations where the content within the document needs to be searched. For those situations, you have the ability to export documents as searchable PDF files. Assuming your repository has full-text search abilities, you can find and retrieve a document by its contents.

To export searchable PDF files to a repository, you simply configure the compiled Create TIFF or PDF ruleset. Figure 9-16 shows the options to select.

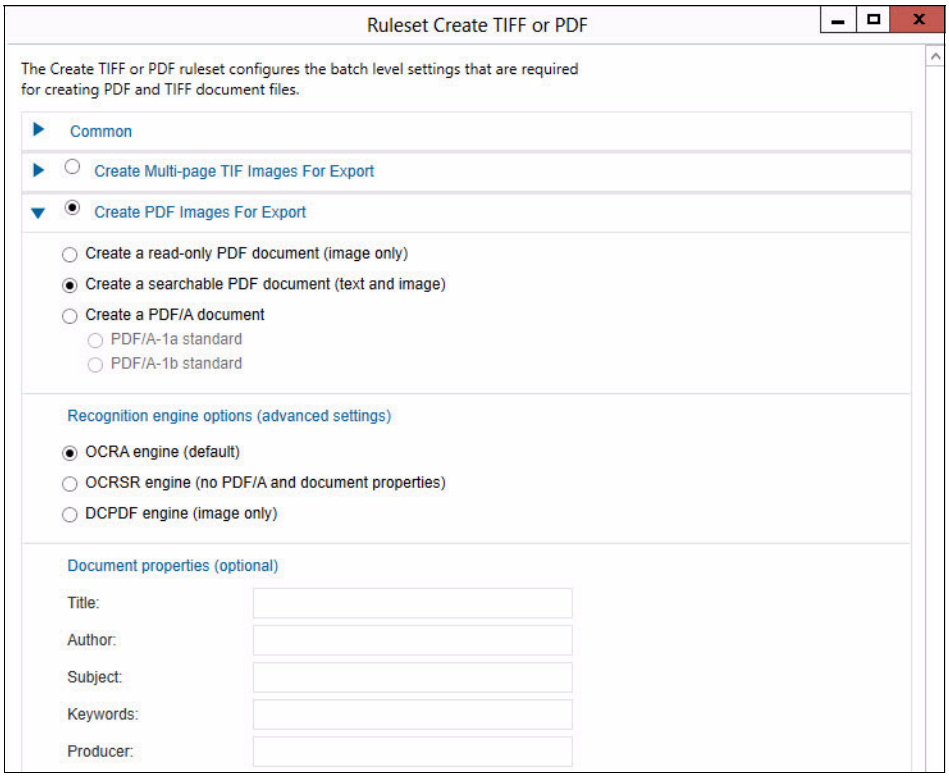


Figure 9-16 Create TIFF or PDF ruleset

The following describes the options to select from Figure 9-16 to create a searchable PDF:

- A searchable PDF file is exported by selecting **Create a searchable PDF document (text and image)**. When creating this type of PDF file, the image used during the capture process is embedded within the PDF file.
- To make the PDF file searchable, all the pages must have optical character recognition (OCR) run on them. The output from this process becomes the searchable text. The OCR engine selected can dictate the PDF format you can create. In this scenario we select the **OCRA engine (default)** option.
- Although not required, you have the option to assign the following document properties:
  - Title
  - Author
  - Subject
  - Keywords
  - Producer

The configuration options selected in Figure 9-16 on page 216 created the searchable PDF in Figure 9-17.

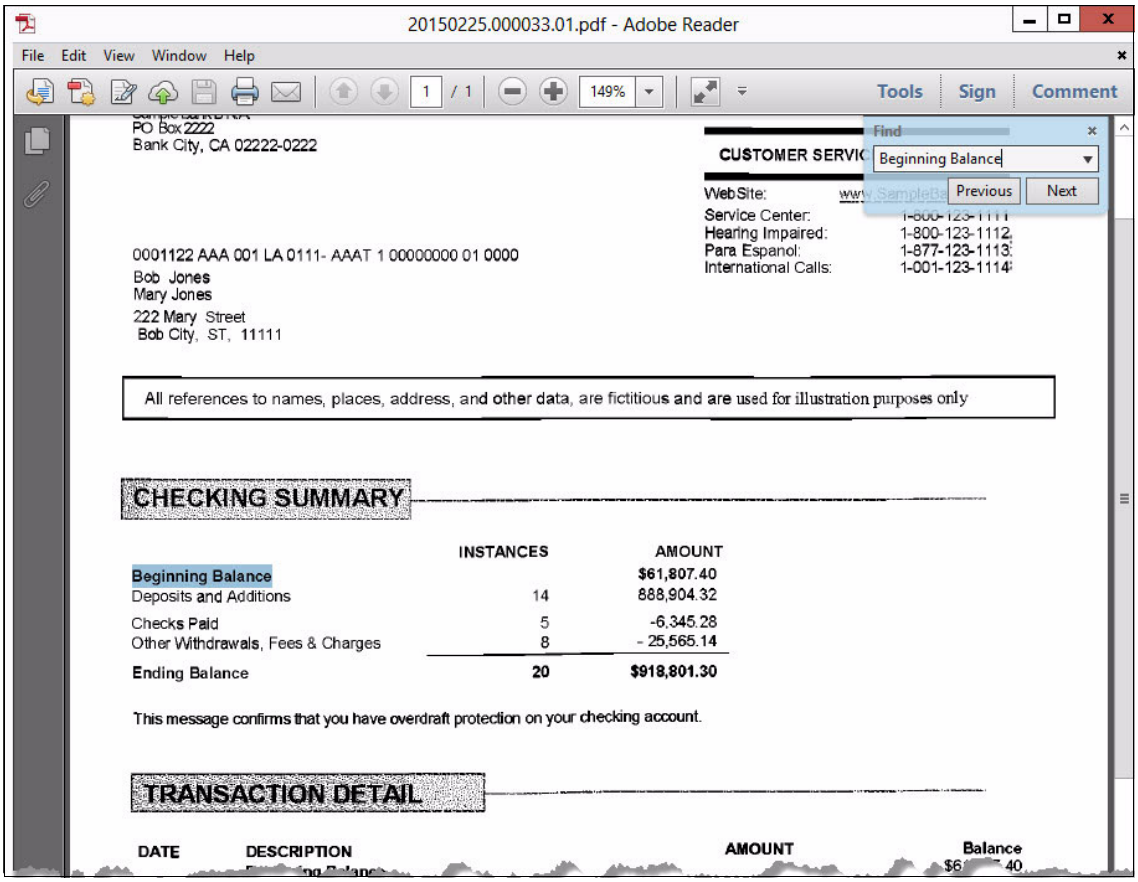


Figure 9-17 Searchable PDF document

> **Note:** The Beginning Balance value, entered in the Find box, is highlighted in the body of the document.

Export a document as a multi-page TIFF file

The pages of any batch can be converted to a multi-page TIFF file. This is a single file that contains multiple images (for example, pages). Commonly, documents and all of their pages are exported as a multiple-page TIFF file. For Datacap to create one, from **Create TIFF or PDF ruleset**, you simply select the **Create Multi-Page TIFF Images for Export** option, as shown in Figure 9-18 on page 218. Depending on factors such as storage availability or determined retrieval performance, you also have the option to compress the TIFF file or use the original image compression.

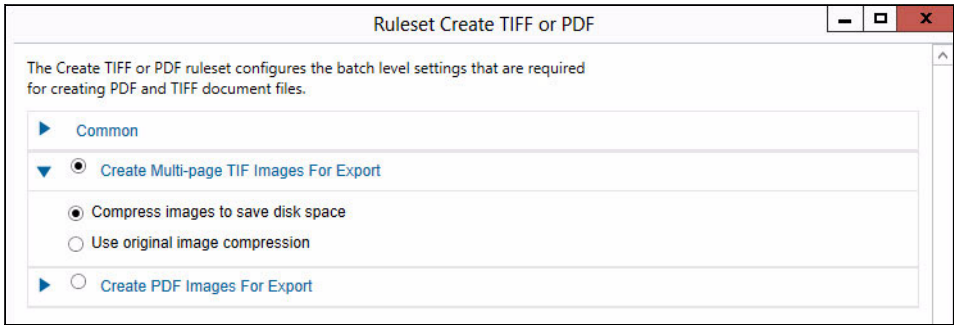


Figure 9-18 Configuration to create a single multi-page TIFF file

When the ruleset in Figure 9-18 is run, a single multi-page TIFF file is created for each document and placed in the documents batch directory.

Figure 9-19 shows the output of the Create TIFF or PDF ruleset with the selected options.

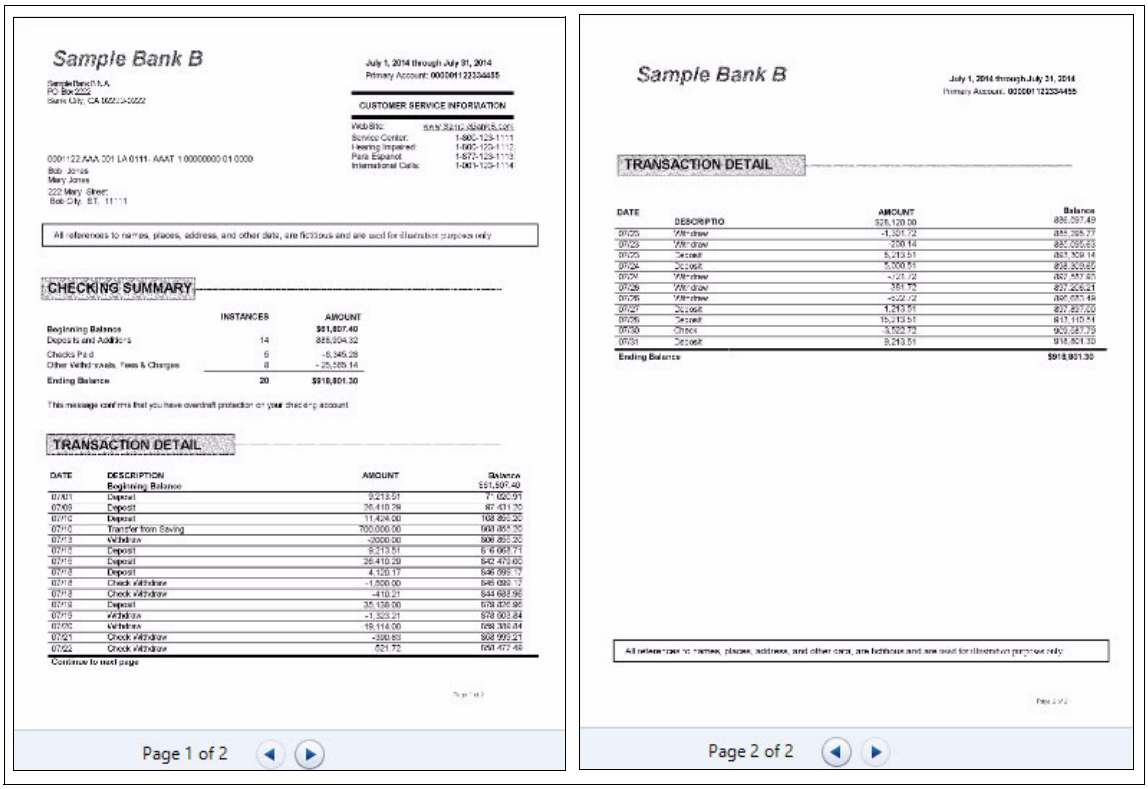


Figure 9-19 Multi-page TIFF file

Export document as JPEG

Beyond multi-page TIFF files, you can convert document pages to a JPEG format. Any of the following file formats can be converted:

- BMP (1, 4, 8, or 24-bit),
- GIF (1, 4, or 8-bit)
- PNG (1, 4, 8, and 24-bit)
- TIFF (1, 4, 8, and 24-bit) with compression (RLE, Group 3 fax, and Group 4 fax, Pack Bits, LZW, JPEG)

Figure 9-20 shows the logic to perform this conversion.

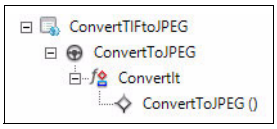


Figure 9-20 JPEG conversion ruleset

The following describes what the functions and actions do, as shown in Figure 9-20:

- The `ConvertIt` function within the ConvertToJPEG rule uses the following action to export page: Export the image as a JPEG (`ConvertToJPEG`).
- This function is attached to the page element in the DCO because we are outputting a single JPEG for each page (see Figure 9-21).

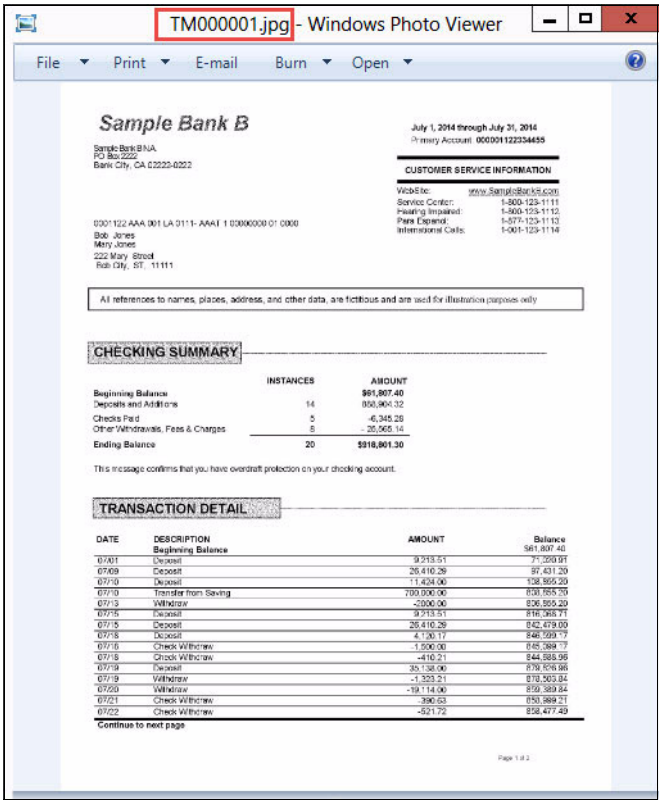


Figure 9-21 Exported JPEG image

General document exporting

There is no limit to the file format that DCO objects can be exported as. Beyond PDF, JPEG, and multi-page TIFF formats, you can also export the original captured file to the repository. If a native action is not provided by Datacap for to meet your export requirements, you can create your own, just as with any other action that is not natively provided.

9.3 Exporting content

After the data is reformatted, and documents are converted, the content is ready to be exported to a repository. A repository can be any system or location content is placed. Datacap provides native integration with the following repositories:

- IBM FileNet P8
- IBM Content Manager
- EMC Documentum
- Microsoft SharePoint
- CMIS content repository
- IBM Content Manager on Demand
- IBM Image Services

In the subsections that follow, we explan exporting content to each of these, beginning with the IBM FileNet Content Manager.

9.3.1 Export content to FileNet Content Manager

The IBM FileNet P8 Connector (P8) integrates Datacap applications with an IBM FileNet Content Engine. You can use the connector to upload documents and index fields into a Content Engine repository. Export configuration options can be set at the batch, document and field levels of the DCO.

Note: To use the Secure Sockets Layer (SSL) to encrypt communications between Datacap and P8, you must set up an SSL-encrypted connection in the FileNet P8 client.

Batch level configuration

Figure 9-22 shows the Export to FileNet Content Manager, batch level, ruleset configuration interface.

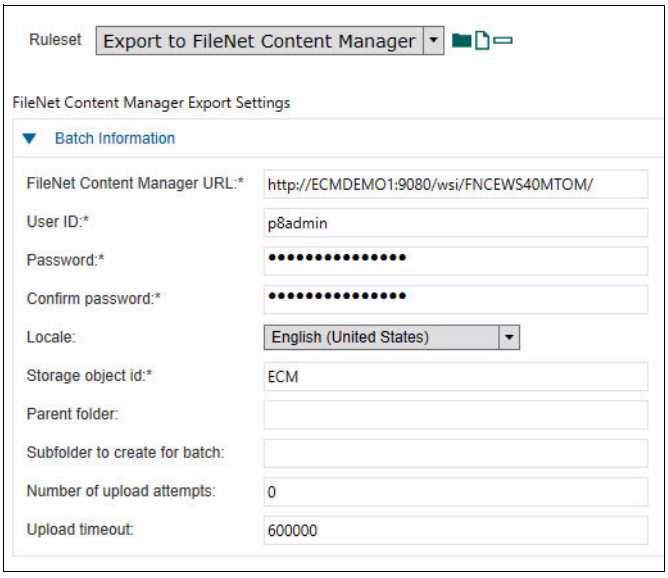


Figure 9-22 FileNet Content Manager batch level configuration interface

The following list identifies each property that is associated with defining a connection to a P8 repository:

- *FileNet Content Manager URL* is a required field. Its value specifies the address of the Content Engine web service.
- *User ID* is a required field. Its value specifies the P8 account the Datacap connector will use to log in to the Content Engine repository.
- *Password* is a required field. Its value is the password associated with the User ID entered in the prior field.
- *Locale* should be set to match the locale of the P8 system.
- *Storage object ID* is a required field. It specifies the identifier for the object store the files will be added to.
- *Parent Folder* is not required. If populated, it identifies the folder into which the document is uploaded in the Content Engine.
- *Subfolder to create for batch* is not a required field. If populated, it identifies the path to a folder into which the document is uploaded in the Content Engine.
- *Number of upload attempts* is not a required field. If populated, it determines the number of times that Datacap will attempt to upload the content before placing the batch in an error state.
- *Upload timeout* is not a required field. If populated, it determines the amount of time to wait for the upload process to complete before placing the batch in an error state.

Note: If the “Parent folder” and “Subfolder to create for batch” fields are empty, the content will be placed in the Unfiled folder.

Document level configuration

Figure 9-23 shows the Export to FileNet Content Manager, document level, ruleset configuration interface.

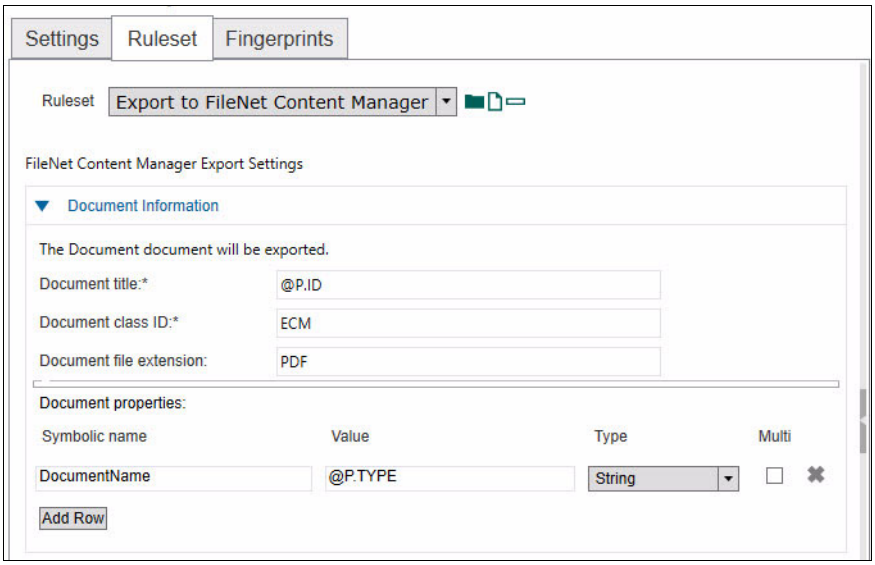


Figure 9-23 FileNet Content Manager document level configuration interface

The following list identifies each property that is associated with exporting document-level property values to a P8 repository:

- *Document Title* is a required field. It determines the name of the document when added to the P8 system.
- *Document class ID* is a required field. It contains the symbolic name of the document class the document will be uploaded into.
- *Document file extension* is an optional field. If populated, it contains the extension for the type of file being uploaded to the P8 system. If left blank, it defaults to a value of `TIF`.
- *Symbolic name*, under Document Properties, represents the internal P8 name of the property. This should be provided by your system administrator.
- *Value*, under Document Properties, represents the value to use to populate the property.
- *Type,* under Document Properties, is a drop-down list of valid property types. The selected type must be able compatible with the property value.
- *Multi* is checked under Document Properties if it is a multi-value property.

Note: Click the **Add Row** button, under document properties, to add as many property values as needed to upload the document.

Field-level configuration

Property values can be set for the document being uploaded from the document and field element within the DCO. Figure 9-24 shows the Export to FileNet Content Manager ruleset configuration interface.

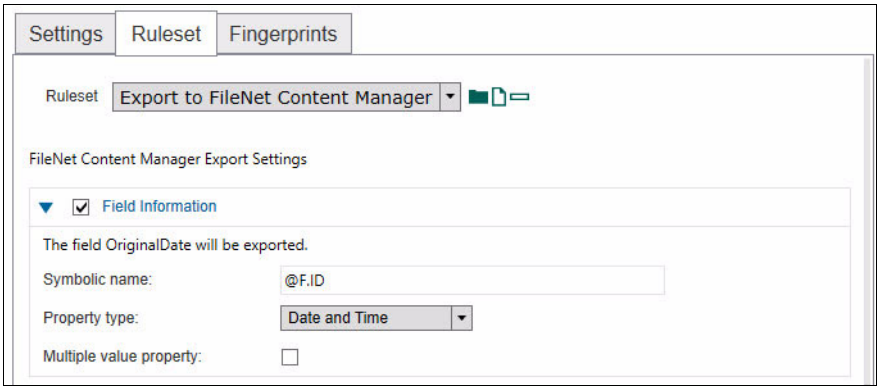


Figure 9-24 FileNet Content Manager field level configuration interface

The following list identifies each property that is associated with exporting field-level property values to a P8 repository:

- *Symbolic name* represents the internal P8 name of the property. This should be provided by your system administrator.
- *Property type* specifies the type of value the property will contain, such as Date and Time or String. The selected type must be able compatible with the property value.
- *Multi* indicates if the specific property is defined as a multi-value property in the P8 system.

Note: The P8 object ID of the uploaded object is written back to the DCO after it is successfully committed to the P8 system.

9.3.2 Export content to IBM Content Manager

The IBM Content Manager Connector ruleset integrates Datacap applications with the IBM Content Manager repository. Use the connector to upload documents and index fields into an IBM Content Manager repository.

Batch level configuration

Figure 9-25 shows the Export to IBM Content Manager ruleset configuration interface.

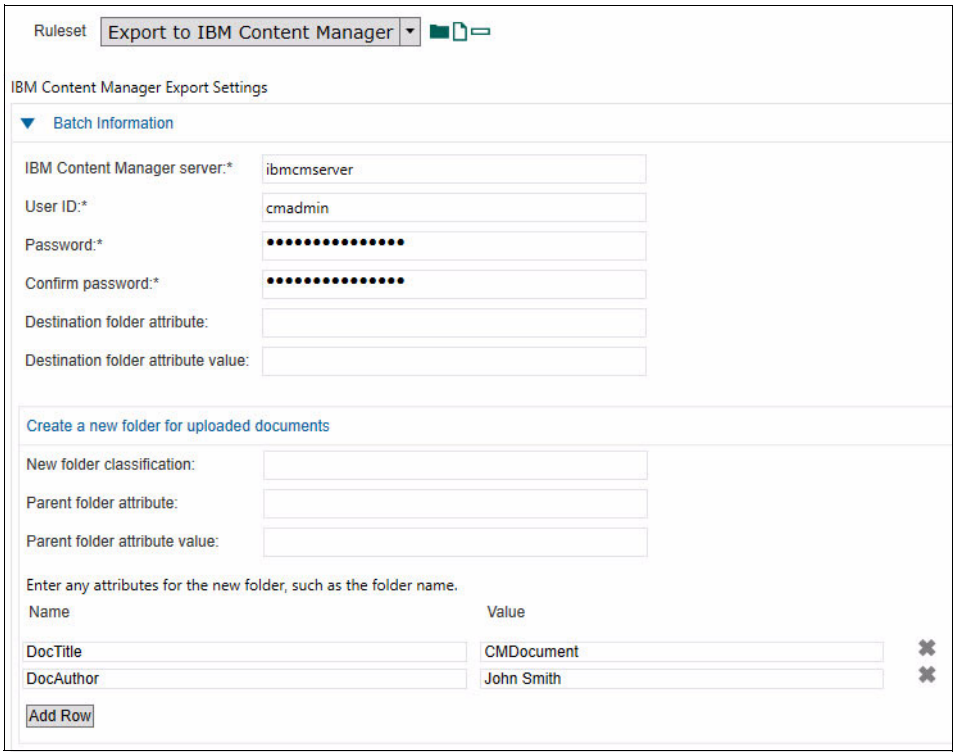


Figure 9-25 IBM Content Manager batch level configuration interface

The following list identifies each property that is associated with defining a connection to an IBM Content Manager repository:

- *IBM Content Manager server* is a required field. Its value specifies the name of the Content Manager server.
- *User ID* is a required field. Its value specifies the Content Manager account the Datacap connector will use to log in to the repository.
- *Password* is a required field. Its value is the password associated with the User ID entered in the prior field.
- *Destination folder attribute* is not required. If populated, it identifies the attribute of the destination folder the document will be uploaded to. Leave this field blank if the destination folder attribute is a GUID or to use the newly created folder as the destination.
- *Destination folder attribute value* is not required. If populated, it identifies the attribute of the destination folder the document will be uploaded to. Leave this field blank to use the newly created folder as the destination.
- *New folder classification* is not a required. If populated, it specifies the classification of the new folder that will be created for the batch. Leave this field blank to skip creating a folder.

- *Parent folder attribute* is not a required field. If populated, it specifies the attribute of the parent for the new folder. Leave this blank if the parent attribute is a GUID.
- *Parent folder attribute value* is not a required field. If populated, it specifies the attribute of the destination folder. Leave this blank if the parent folder GUID is entered.
- *Name* in Folder Attribute is not required. If populated, the symbolic name of the property within the Content Manager system is entered.
- *Value* in Folder Attribute is not required. If populated, enter the value that is associated with the property name specified.

Note: Under the Enter Attributes section, click **Add Row** to add as many property values as needed to upload the document.

Document level configuration

Figure 9-26 shows the Export to IBM Content Manager ruleset configuration interface.

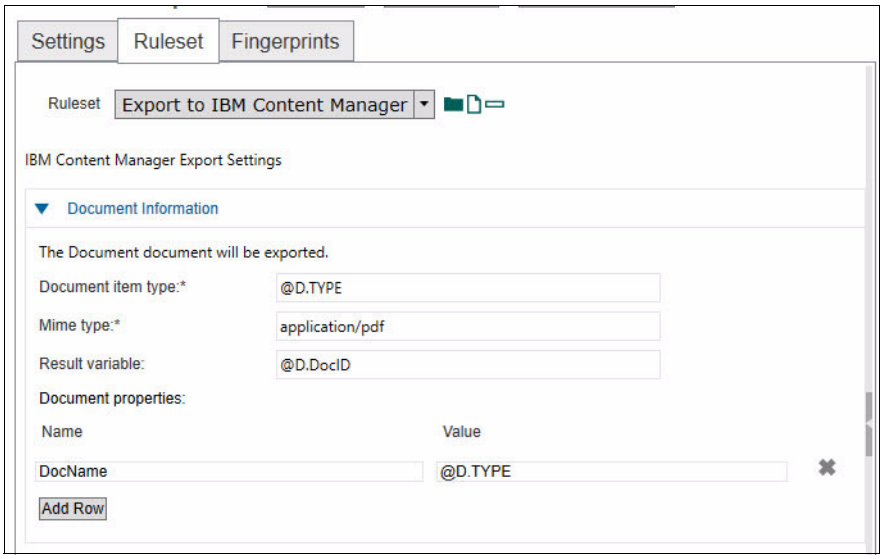


Figure 9-26 Export to IBM Content Manager ruleset configuration interface

The following list details the configuration settings that are assigned to a document when it is uploaded to a Content Manager repository:

- *Document item type* is a required field. It sets the document name for the uploaded document.
- *Mime type* is a required field. It identifies the type of document being uploaded. Figure 9-27 shows examples.

Figure 9-27 Example MIME type values

- *Result variable* is an optional field. If populated, it specifies the Datacap DCO object the document ID value is written to.
- *Name* represents the symbolic name of the property that you want to configure.
- *Value* represents the value to use to populate the specified property.

Note: Under document properties, use **Add Row** to add as many property values as you need to upload the document.

Field level configuration

Property values can be set for the document being uploaded from the document and field element within the DCO. Figure 9-28 shows the Export to IBM Content Manager ruleset configuration interface.

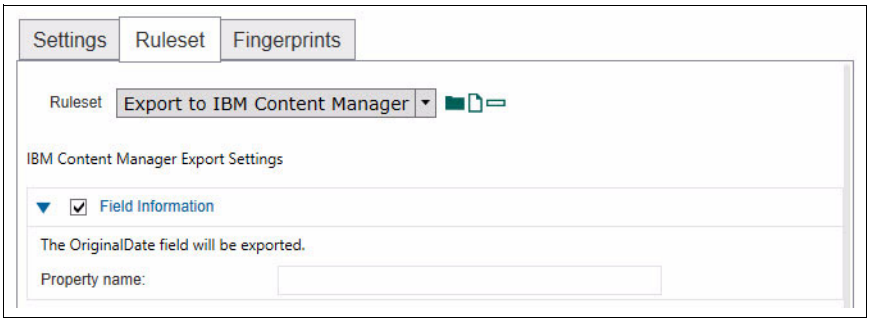


Figure 9-28 IBM Content Manager field level configuration interface

The following identifies each configuration setting that is associated with exporting field level property values:

- *Property* is not required.
- If populated, it specifies the property name where the DCO field value will be stored.

9.3.3 Export content to Documentum

The Documentum Connector actions integrate Datacap applications with the Documentum Docbase content repository. You can then use the Documentum Connector actions to upload documents and index fields into a Documentum repository.

Follow this procedure for connecting and uploading a document:

1. Log in to the Documentum repository.
2. Specify the content type or format in which to release documents to the Documentum repository, such as TIFF or PDF.
3. Set the name of the folder in Datacap from which to upload the documents into the Documentum repository.
4. Set the object name for the file that is uploaded into the Documentum repository.
5. Upload the indexed documents or pages into the Documentum repository.

Figure 9-29 shows the Documentum actions library.

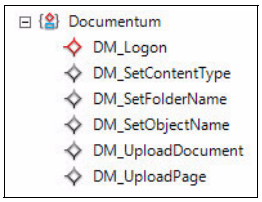


Figure 9-29 Documentum actions library

Figure 9-30 details how to upload a page from a Datacap batch to a Documentum repository.

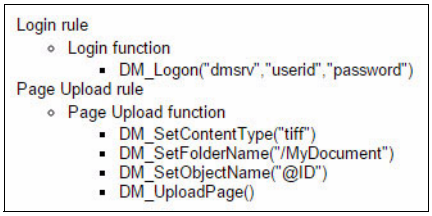


Figure 9-30 Documentum upload example

The functions and actions shown in Figure 9-30 complete the following tasks:

- The `Login` function within the Login rule uses the following action to open a connection to the Documentum repository:
  - Specify the domain name, server name, user ID, and password, which Datacap then uses to log in to the repository (`DM_Login`).
- The `Page Upload` function within the Page Upload rule uses the following actions to upload a page to the Documentum repository:
  - Sets the content type in the repository for the object, for example, TIFF, JPEG, DOC (`DM_SetContentType`).
  - Specify the name of the Documentum folder where Datacap places the uploaded file (`DM_SetFolderName`).
  - Sets the name of the file that you are uploading as it appears in the Documentum repository (`DM_SetObjectName`).
  - Upload the selected page from the document (`DM_UploadPage`).

Note: The `DM_UploadDocument` action uploads all of the pages that are attached to a document. An XML file called `DM_Uploaded.xml` is created in the batch directory. This file lists all of that pages that have been uploaded.

9.3.4 Export content to Microsoft SharePoint

The Datacap Connector for Microsoft SharePoint actions integrate Datacap applications with Microsoft Office SharePoint Services for Microsoft SharePoint. You can use SharePoint Connector actions to upload documents and set index fields in a SharePoint library.

Follow this procedure for connecting and uploading a document:

1. Log in to the Microsoft SharePoint library.
2. Identify and set up the URL of the SharePoint library.
3. Specify the content type that defines the fields within a document library for the uploaded documents, such as an Invoice.
4. Set the format in which to release documents to the SharePoint library, such as TIFF or PDF.
5. Create a folder in the SharePoint into which you upload documents.
6. Set the column properties (index values) in SharePoint for the documents that you want to upload.
7. Upload the indexed documents into the SharePoint library.

Figure 9-31 shows the Datacap Microsoft SharePoint (SP) library actions.

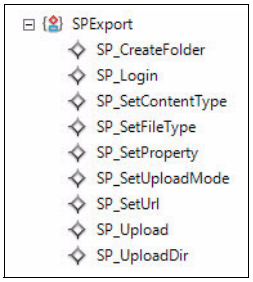


Figure 9-31 MS SharePoint library actions

Figure 9-32 shows the details for how to upload a document from a Datacap batch to a SharePoint repository.

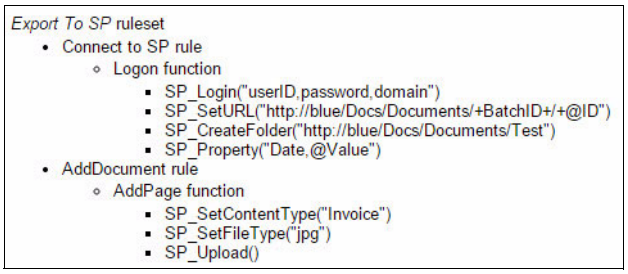


Figure 9-32 Upload document to SharePoint ruleset

The functions and actions in Figure 9-32 complete these tasks:

- The `Login` function within the Connect to SharePoint (SP) rule uses the following actions to open a connection to the SP repository:
  - Set the User ID, password, and optional SharePoint domain (`SP_Login`).
  - Specify the URL address of the SharePoint library (`SP_SetURL`).
  - Set the folder in the SharePoint library into which your documents will be uploaded to (`SP_CreateFolder`).
  - ID the column property in SharePoint for the documents that you want to upload (`SP_Property`).

- The `AddPage` function within the `AddDocument` rule uses the following actions to upload a page to the SP repository:
  - The name of the content type that defines the fields within a document library, such as an Invoice (`SP_SetContentType`)
  - Defines the format in which to upload the document to the SharePoint library, for example TIFF or PDF (`SP_SetFileType`)
  - Uploads the image file and any indexes specified for the current page, document, or batch to SharePoint (`SP_Upload`)

Note: If some documents in a batch are successfully uploaded and some fail, and the batch is re-run through the SharePoint Upload task, only documents that failed to upload will be re-uploaded.

9.3.5 Export content for upload to IBM Content Manager OnDemand

You can configure Datacap to export index data and files into Content Manager OnDemand. The Content Manager OnDemand import tool contains the ARSLOAD component. ARSLOAD can accept a flat index file that contains index data and locations of files that can be uploaded. You can use the generic Datacap Export library actions to create an index file in a format supported by ARSLOAD.

For detailed information about the ARSLOAD file formats and how to configure Datacap to export data, see the IBM Technote titled “IBM Datacap Taskmaster Capture export to Content Manager On Demand (CMOD)”:

http://www.ibm.com/support/docview.wss?uid=swg21502807

9.3.6 Export content to FileNet Image Services repository

The main function of the FileNet Image Services Connector actions is to upload documents and commit images to an IBM FileNet Image Services (IS) library:

Follow these steps to connect and upload a document:

1. Access and open an IBM FileNet Image Services library.
2. Create a FileNet document to upload into the library.
3. Define an Index Map that links FileNet properties to values that are associated with objects of the Document Hierarchy.
4. Associate images with FileNet documents.
5. Upload indexed documents and images for commitment to the library.

Figure 9-33 shows the Datacap FileNet IS Library actions.

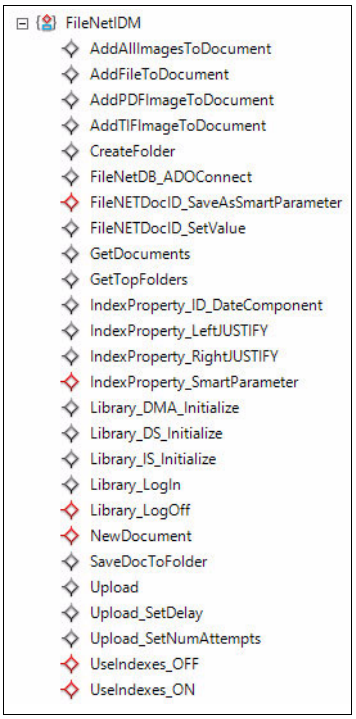


Figure 9-33 FileNet Image Services actions library

Figure 9-34 shows a ruleset that logs on to FileNet Image Services and uploads a single page document into the library.

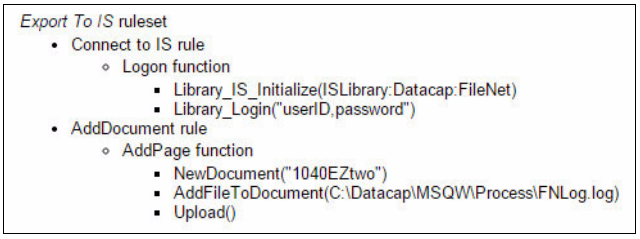


Figure 9-34 Upload file to IS ruleset

The following list describes what each function and action, shown in Figure 9-34, does:

- The `Logon` function within the Connect to IS rule uses the following actions to open a connection to the IS repository:
  - Initialize the connection to the IS library (`Library_IS_Initialize`).
  - Log in to the initialized IS library by using the user ID and password (`Library_Login`).
- The `AddPage` function within the AddDocument rule uses the following actions to upload a file to the IS library:
  - Define a new document and assign it a FileNet Document Class (`NewDocument`).
  - Add the specified file to the document created in the prior step (`AddFileToDocument`).
  - Commit the active document to the initialized IS library (`Upload`).

Note: The `IndexProperty_SmartParameter` action in the FileNetIDM library can be used to populate the index values of the specified document class.

9.3.7 Export content to CMIS repository

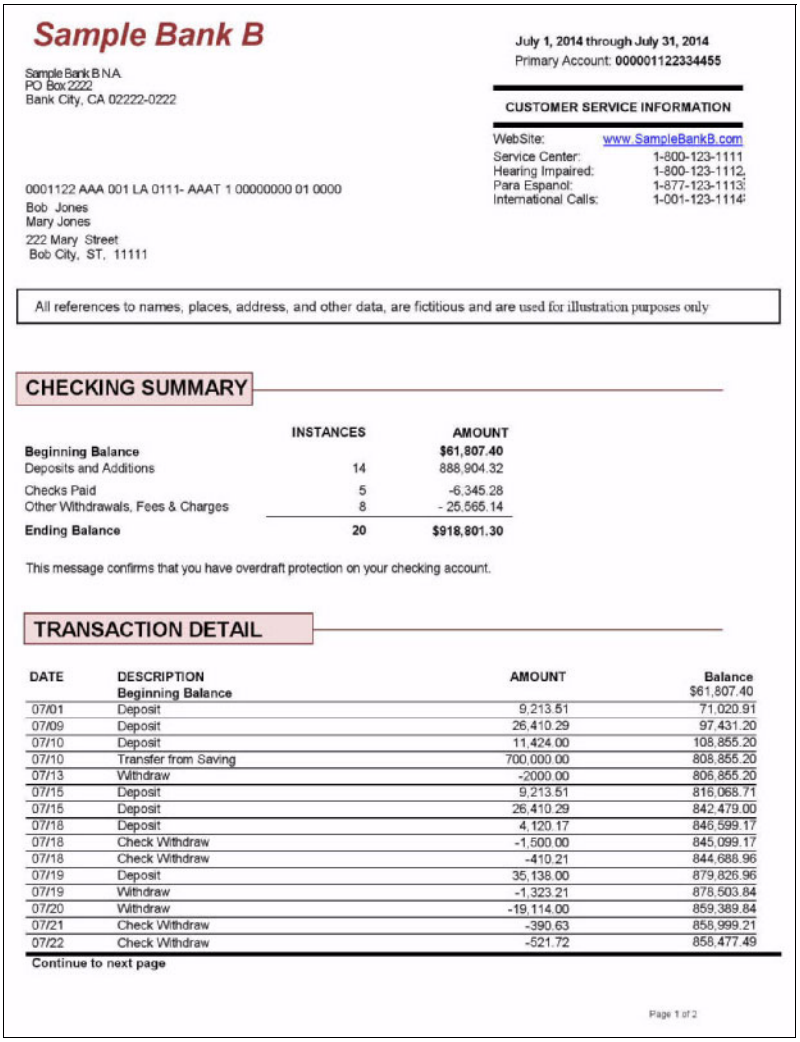
IBM Content Management Interoperability Services (CMIS) is an open standard that enables communication between Datacap applications and CMIS-compliant repositories.

This section uses the data lists in Table 9-4 and the CMISClient action library shown in Figure 9-35 to demonstrate exporting content to a CMIS repository. Specifically, we export to an IBM FileNet Content Manager repository.

Table 9-4 Sample data for upload to a CMIS repository

Field name	Captured value
AccountNo	1122334455
CustomerName	Bob Jones
StatementDate	07/01/2014

The CMISClient action library enables you to access the CMIS repository, set document attributes, create folders, and upload documents to the server for storage, among multiple other actions.



Sample Bank B

Sample Bank B N.A.
PO Box 2222
Bank City, CA 02222-0222

July 1, 2014 through July 31, 2014
Primary Account: 000001122334455

CUSTOMER SERVICE INFORMATION

WebSite:	www.SampleBankB.com
Service Center:	1-800-123-1111
Hearing Impaired:	1-800-123-1112
Para Espanol:	1-877-123-1113
International Calls:	1-001-123-1114

0001122 AAA 001 LA 0111- AAAT 1 00000000 01 0000
Bob Jones
Mary Jones
222 Mary Street
Bob City, ST, 11111

All references to names, places, address, and other data, are fictitious and are used for illustration purposes only

CHECKING SUMMARY

	INSTANCES	AMOUNT
Beginning Balance		**$61,807.40**
Deposits and Additions	14	888,904.32
Checks Paid	5	-6,345.28
Other Withdrawals, Fees & Charges	8	- 25,565.14
Ending Balance	**20**	**$918,801.30**

This message confirms that you have overdraft protection on your checking account.

TRANSACTION DETAIL

DATE	DESCRIPTION	AMOUNT	Balance
	Beginning Balance		$61,807.40
07/01	Deposit	9,213.51	71,020.91
07/09	Deposit	26,410.29	97,431.20
07/10	Deposit	11,424.00	108,855.20
07/10	Transfer from Saving	700,000.00	808,855.20
07/13	Withdraw	-2000.00	806,855.20
07/15	Deposit	9,213.51	816,068.71
07/15	Deposit	26,410.29	842,479.00
07/18	Deposit	4,120.17	846,599.17
07/18	Check Withdraw	-1,500.00	845,099.17
07/18	Check Withdraw	-410.21	844,688.96
07/19	Deposit	35,138.00	879,826.96
07/19	Withdraw	-1,323.21	878,503.84
07/20	Withdraw	-19,114.00	859,389.84
07/21	Check Withdraw	-390.63	858,999.21
07/22	Check Withdraw	-521.72	858,477.49

Continue to next page

Page 1 of 2

Figure 9-35 Sample document for upload to a CMIS repository

Figure 9-36 shows the CMIS Client Library actions.

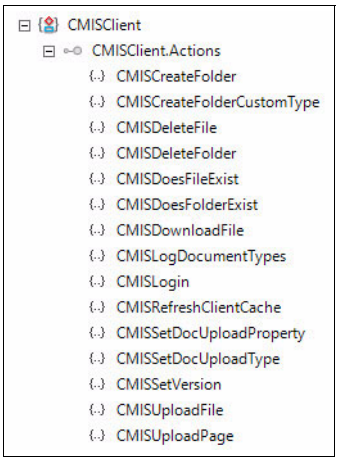


Figure 9-36 Datacap CMIS action library

Figure 9-37 shows a ruleset that logs on to the CMIS repository to create a folder and an index, and then upload a page with the batch.

Figure 9-37 Upload to a CMIS repository ruleset

The following list describes what each function and action, shown in Figure 9-37, does:

- The *OpenIt function* within the OpenCMISConn rule uses the following actions to open a connection to the CMIS repository:
  - Log in to the repository with the specified credentials (`CMISLogin`).
- The `CreateIt` function within the CreateExportDir rule checks whether a folder within the repository exists. If it does not, one is created:
  - Check the repository to see if a specific folder exists (`CMISDoesFolderExist`).
  - Create a folder to upload a page to (`CMISCreateFolder`).
- The `ExportIt` function within the ExportPage rule uses the following actions to define the type of page being uploaded and populate the associated property values:
  - Set the version of the file that will be uploaded (`CMISSetVersion`).
  - Set the type of file that will be uploaded (`CMISSetDocUploadType`).

– Set the value for the specified property associated with the file type being uploaded (`CMISSetDocUploadProperty`). This action is called for the three property values that are being set (`AccountNo`, `CustomerName`, `StatementDate`).

– Upload the current DCO page to the repository (`CMISUploadPage`).

Figure 9-38 is the IBM Content Navigator interface. It shows where the file was uploaded to (upper left), the file reference (center column), and values assigned to the properties (lower-right). The version of the document is also shown in the lower-right area.

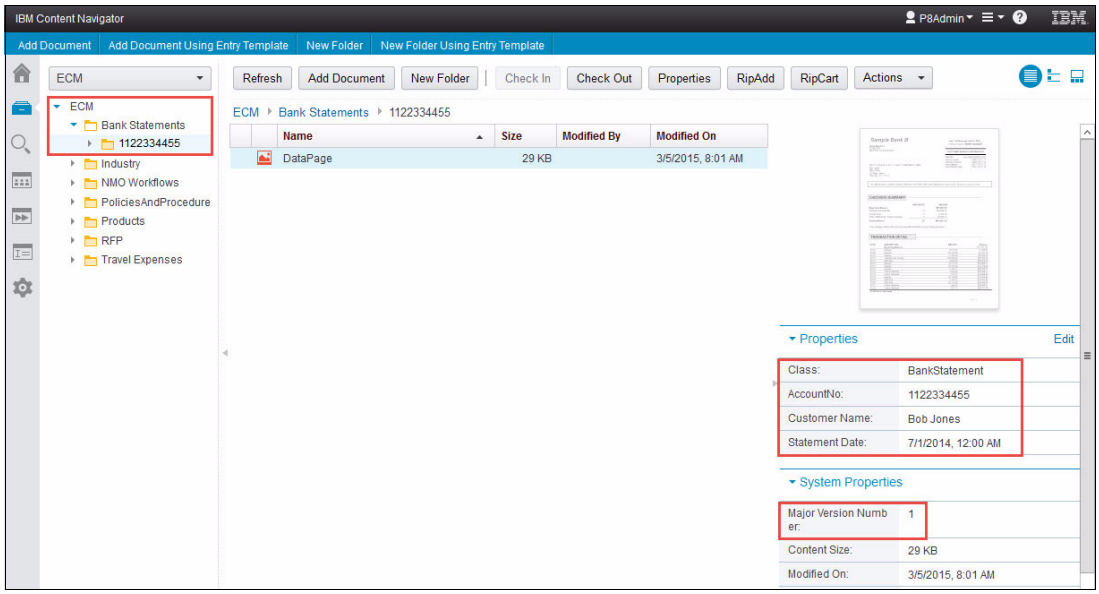


Figure 9-38 IBM Content Navigator showing the uploaded file and attributes

For more information about CMIS and IBM support for it, see the Content Management Interoperability Services (CMIS) web page:

http://www.ibm.com/software/ecm/cmis.html

9.3.8 Access control to content repositories

To use the Datacap Connector Actions to upload documents into the repository, you must have write access to a folder on the repository and privileges to create and view documents in that folder.

Access control is handled differently by each of the repositories and their connectors:

- For Datacap Connector for IBM Content Manager, access is controlled through the IBM Content Manager authentication.
- For Datacap Connector for FileNet Content Manager, access is controlled through the IBM FileNet Content Manager authentication.
- For Documentum Connector, authentication is done by using the Login action with user credentials that are managed by Documentum.
- For Datacap Connector for Microsoft SharePoint, authentication is done by using the Login action with user credentials that are managed by SharePoint.
- For Datacap Connector for FileNet Image Services, authentication is done by the library into which you are importing the documents.

10

Datacap user experience in IBM Content Navigator

Datacap Navigator, a web user interface, is a new capability in IBM Datacap version 9.0. Based on the IBM Content Navigator technology, Datacap Navigator delivers an improved user experience in a rich and responsive client that is familiar to users and consistent with other IBM Enterprise Content Management products.

This chapter introduces Datacap Navigator functions, takes you through the user experience for each main function, and describes how to configure the Datacap Navigator user interfaces. It includes the following sections:

- Introduction to Datacap Navigator
- User experience
- Configuring an application

10.1 Introduction to Datacap Navigator

With Datacap Navigator, users can scan, upload, classify, and verify documents. Users can save their preferred panel layouts of the Scan, Upload, Classify, and Verify tasks. The viewer can be split off to a separate window and displayed on a separate monitor to improve productivity and ease of use.

Supervisors can use the Job Monitor to manage work. They can display the thumbnail of the first page of the batch, and then filter and sort jobs and select and reorder the columns, including extra custom batch fields.

Administrators can configure users, groups, workstations, tasks and jobs, shortcuts, and functional security. A graphical panel designer configures the Verify data entry panels and start batch panel.

Datacap Navigator simplifies deployment of lookups and validation features in conjunction with other Enterprise Content Manager applications, such as Case Manager. For lookup and verification, Datacap Navigator uses the data connector capability of IBM Content Navigator External Data Services.

As illustrated in Figure 10-1, Datacap Navigator is a plug-in component that operates within Content Navigator. To access Datacap, Content Navigator accesses the Datacap Windows Service (formerly wTM). It runs as a Microsoft Windows service that presents RESTful web services endpoints that display the Datacap capabilities.

Datacap applications are implemented as Content Navigator repositories. Repositories are displayed in Content Navigator Desktops to users and supervisors. Content Navigator Desktops can combine features and repositories from Datacap with other Enterprise Content Manager products. The result is a single, combined Enterprise Content Manager user interface for capture, case management, browsing, search, and records.

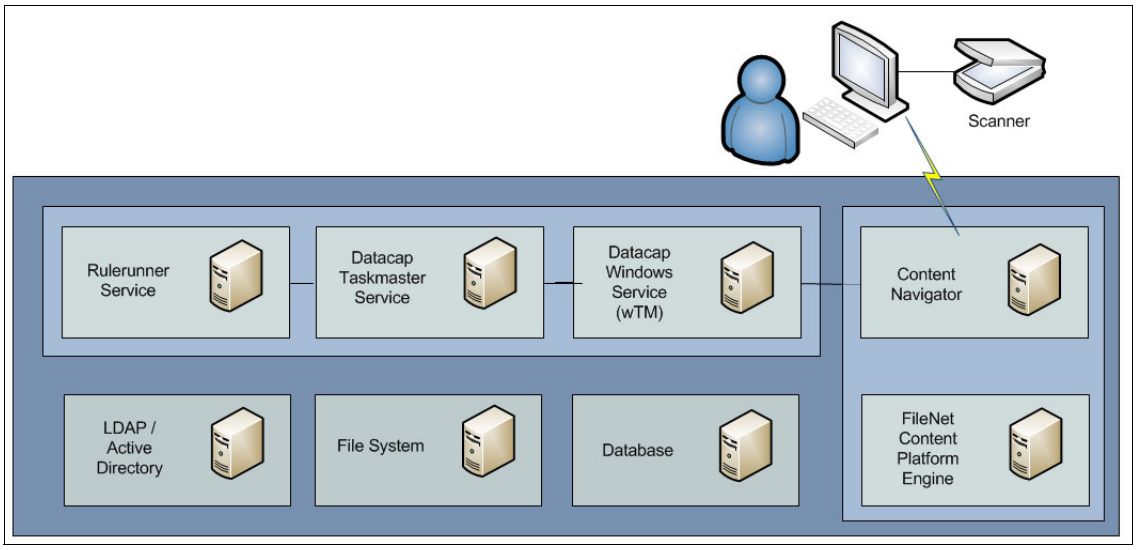


Figure 10-1 Datacap Navigator overview

10.2 User experience

Figure 10-2 shows the manual login panel. To maintain consistency with other areas of Content Navigator, the user does not enter a station number. The station number is set in the user's setting rather than in the login panel.

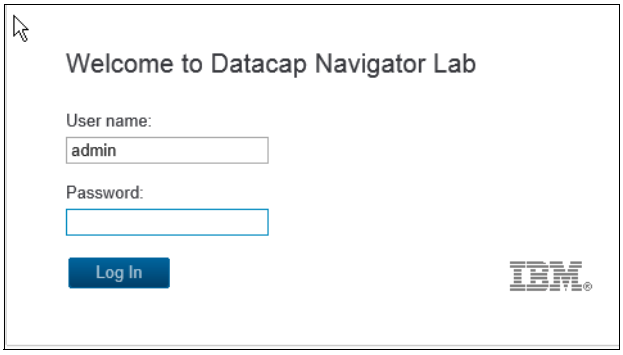


Figure 10-2 Login panel

After the user is logged in, a web page displays the Content Navigator Desktop. The Datacap Navigator desktop is embedded in Content Navigator and uses that user interface. Datacap Navigator can also be configured to provide access to other features of Content Navigator, such as the ability to browse repositories, to allow users to switch easily between functions within the same familiar interface, including Datacap, as shown in Figure 10-3 on page 236. For a particular desktop, you can configure which sections to display, including these three primary sections:

Job Monitor A list of the batches that are being processed by Datacap.

Shortcut panel A list of all of the Datacap shortcuts. This gives the user a list of all of the tasks that they are authorized to run. Optionally, a quick launch panel can also display a single icon for each major type of task. This gives users a quick and easy list to launch tasks.

Feature list A list of all of the features configured for the Content Navigator Desktop. These features can include features of Datacap and other Enterprise Content Manager products. The example shows icons that link to the following features: Datacap View, Datacap Administration View, and Repository browsing (Enterprise Content Manager repository).

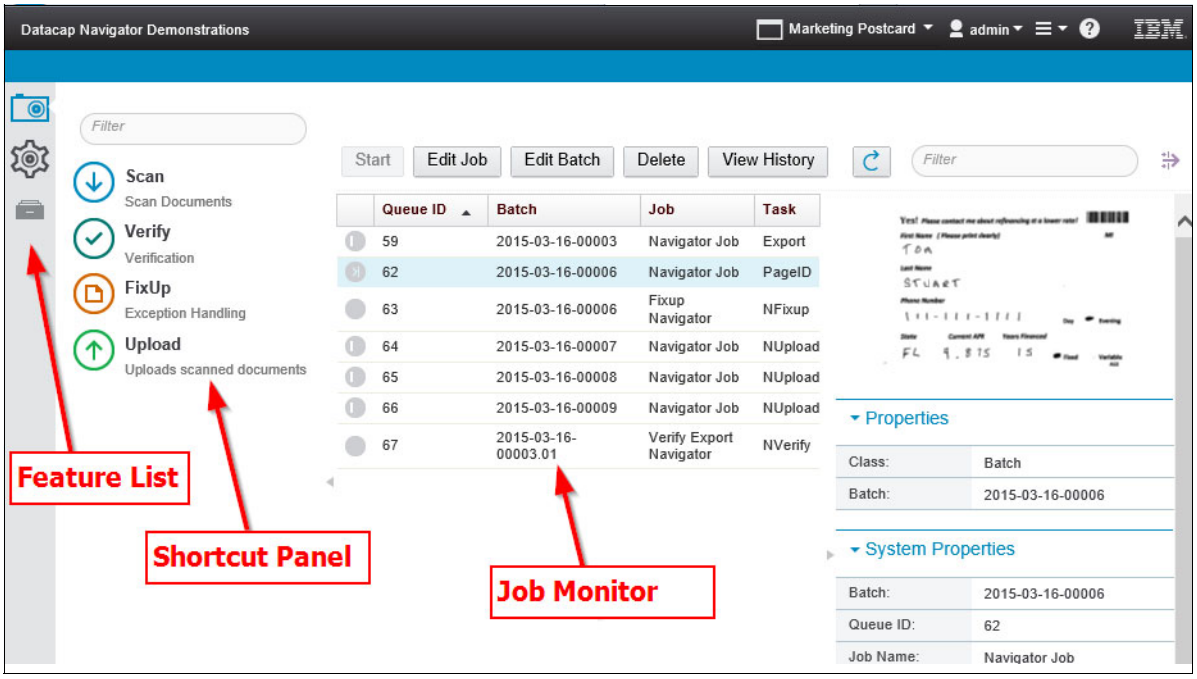


Figure 10-3 Content Navigator Datacap view

10.2.1 Job Monitor

Datacap Navigator supports operational management of the Datacap system by using the Job Monitor. Access to the Job Monitor is controlled through access privileges so that only authorized users have access to it. Figure 10-3 shows a list of job monitors. When a batch is selected, a detail view displays that includes a thumbnail image of the first page of the batch and the batch properties.

You can run user interface tasks by clicking **Start** or by double-clicking a task in the list. You cannot run background tasks directly from Datacap Navigator. Instead, they are run in the Datacap Rulerunner Server. If you select a background task, the task properties display rather than running the task.

With sufficient privileges, a user can edit job and batch properties and can batches. The View History button shows the list of tasks that have run previously for the selected batch. A filter helps you find batches with specific properties. This is useful when the system is processing many batches.

Scanning

One of the primary functions of Datacap is scanning paper documents. Datacap Navigator supports the operation of scanners from the web browser. Scanners that use an industry TWAIN driver can be operated.

The scan task panel can display three sections:

Scanned Pages The image viewer that can display a page and thumbnails
Batch Structure A list of scanned pages
Start Panel A panel to enter data that applies to the entire batch

After scanning or selecting pages from local files, the images are displayed in a list on the Batch Structure section. You can click pages and display full-size images in the Image Viewer. Figure 10-4 shows the scan task with the Scanned Pages and Batch Structure sections.

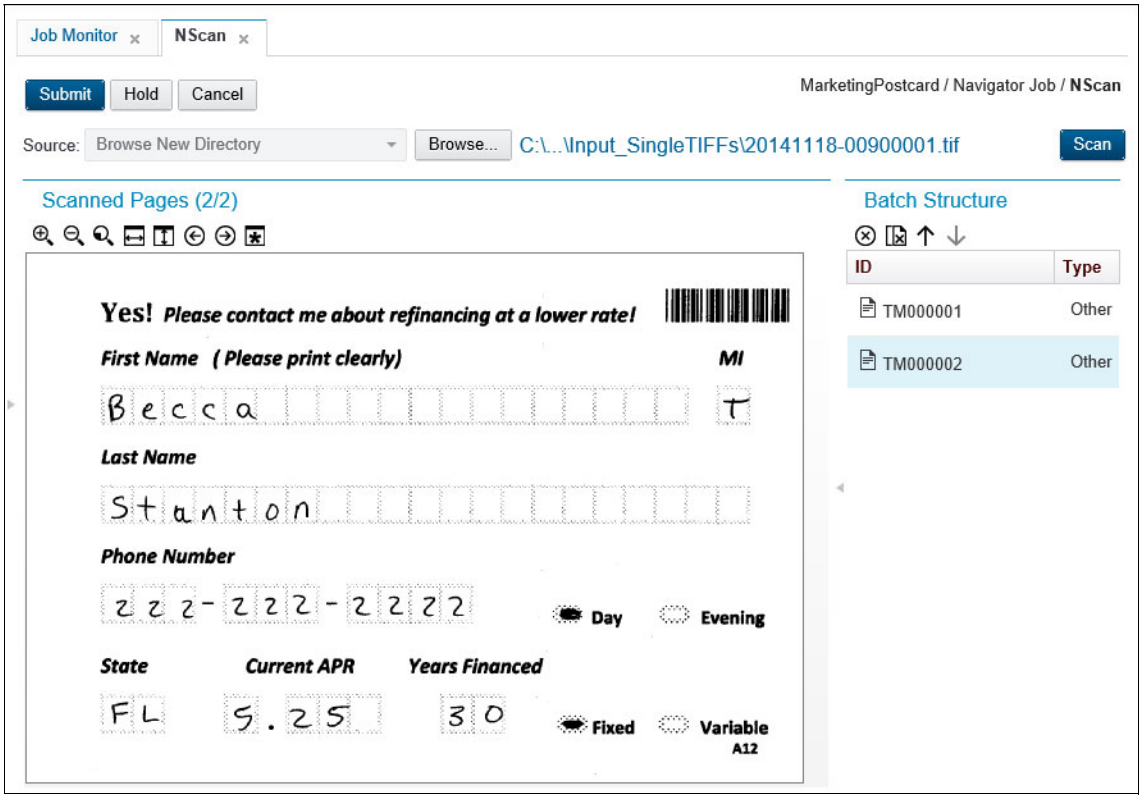


Figure 10-4 Scan task

You can also show a list of thumbnail images by double-clicking the image, as shown in Figure 10-5. Clicking a thumbnail image to navigates to a page in the Batch Structure list. Clicking **Submit** completes the scan. The scan task can be configured to upload the images immediately or in a separate Upload task.

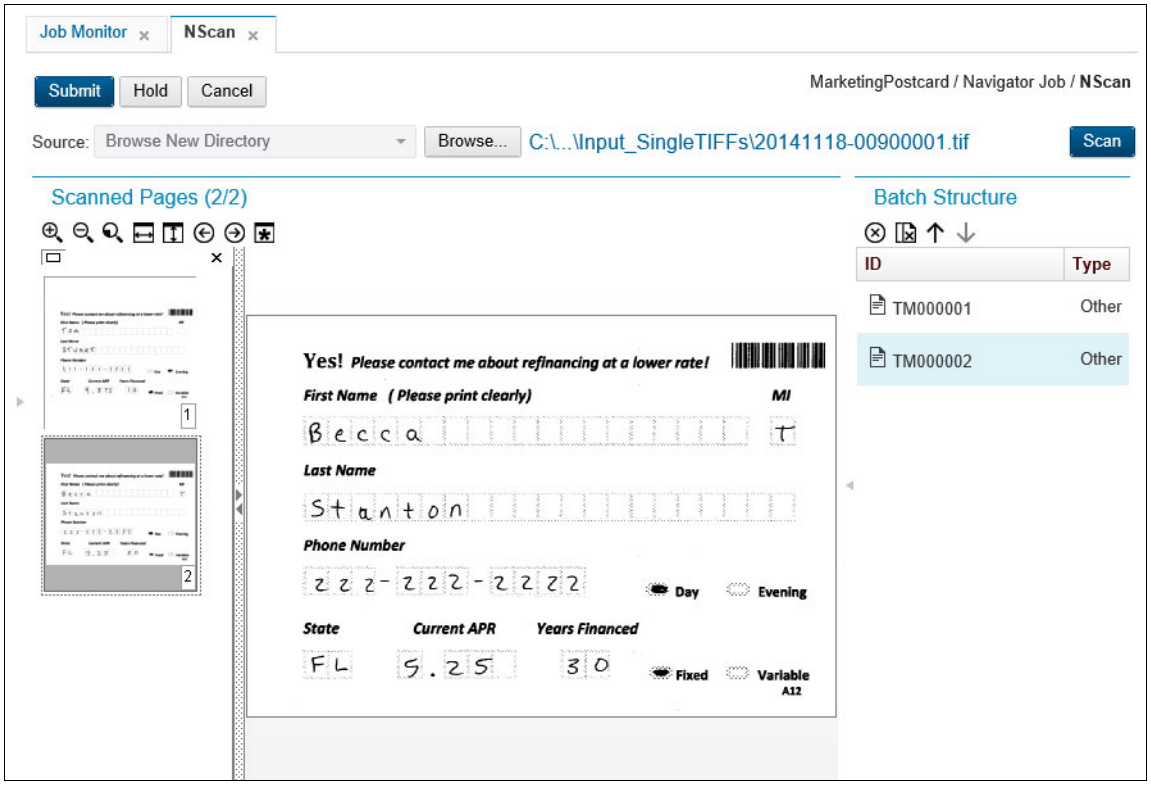


Figure 10-5 Scan task with thumbnails

10.2.2 Classify

In the same manner as other processes in Datacap, the batch is processed in the background on a Datacap Rulerunner Server that identifies pages and assembles documents. If a batch is found to have an invalid structure, it can be routed to a fix-up task. For example, a batch with missing or mis-ordered pages could have an invalid structure. Datacap Navigator includes a user interface, `Classify.js`, for fix up. It can change the page and document types, reorder the pages, merge and split documents, and mark documents and pages for deletion. The fields in the Batch Structure section can be modifiable. Pages and documents can be rearranged by dragging or by using the buttons on the toolbar.

There are two sections on the Classify page, which are shown in Figure 10-6:

Image Viewer Displays the selected page and thumbnails
Batch Structure A list of scanned pages

The fields in the Batch Structure section can be modifiable. Pages and documents can be rearranged by dragging or by using the buttons on the toolbar.

The Field Details panels are dynamically generated by the system and require no additional setup. However, if you want to create your own layout, you can do this with a custom panel. You can rearrange the fields and change the appearance and behavior of the panel in a variety of ways.

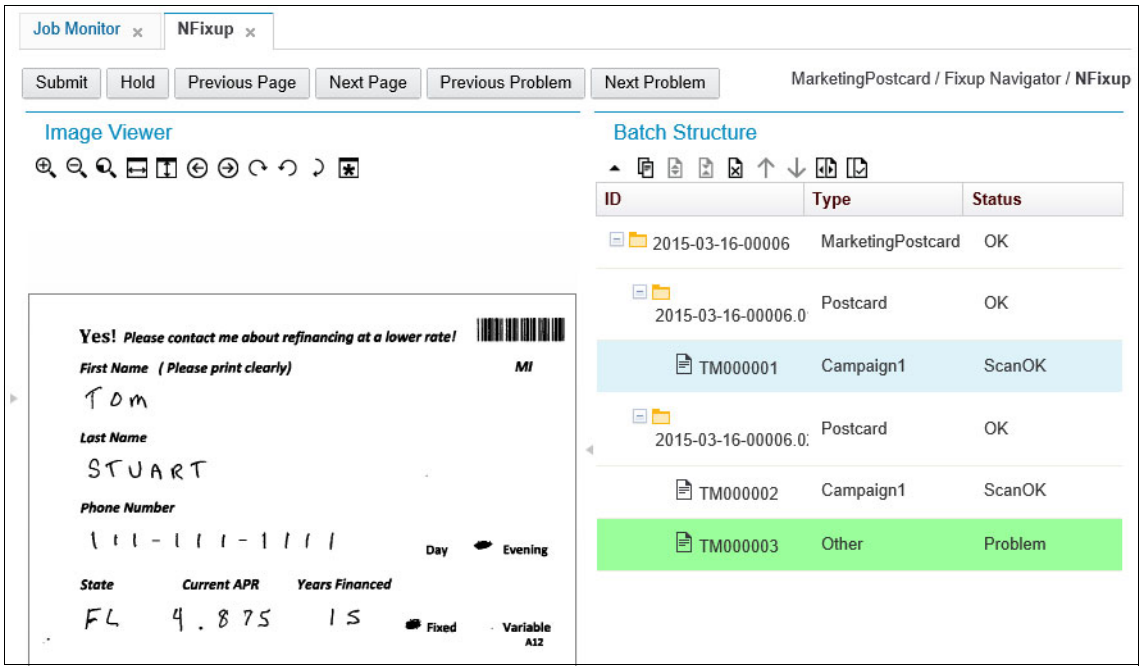


Figure 10-6 Classify task page

10.2.3 Verification

Datacap Rulerunner also validates and flags documents, pages, and fields that need human review. Datacap Navigator includes a user interface, `Verify.js`, for verification where a person reviews the batch to enter data and make corrections. Users can also change the page and document types, reorder the pages, merge and split documents, and mark documents and pages for deletion. The three sections on the Verify page are shown in Figure 10-7:

Field details A data entry panel for entering and correcting fields
Image Viewer Displays the selected page and thumbnails
Batch Structure A list of scanned pages

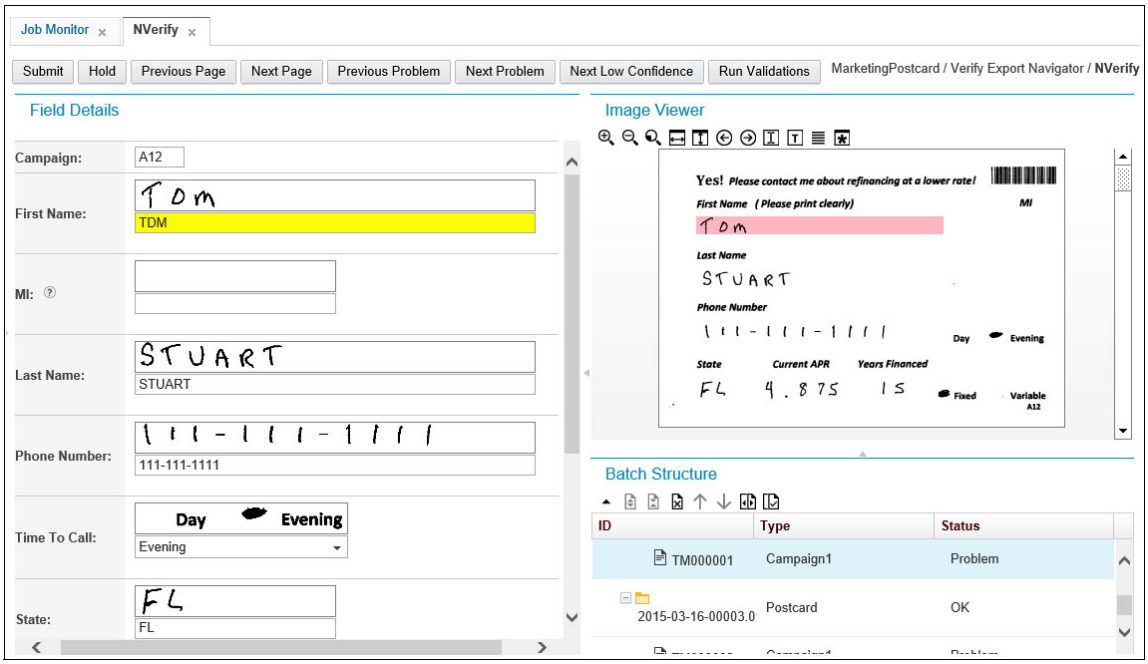


Figure 10-7 Verification task

10.2.4 User settings

Each user can make changes to settings that affect the appearance and operation of the user interface. To access the setting, select the drop-down option on the user name (admin) at the upper right corner of the window where the user can change their individual settings (see Figure 10-8). This opens the Settings panel where there is a separate tab for each type of task, global settings, and settings for the job monitor and task lists. The station ID is set in the Global Settings tab.

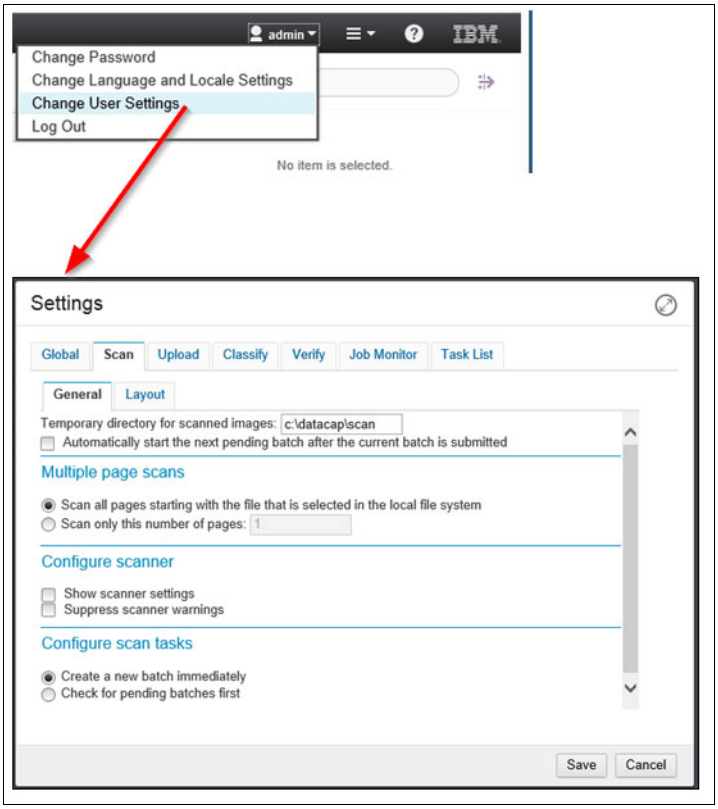


Figure 10-8 Changing user settings

10.3 Configuring an application

This section introduces how to configure an application to use the Content Navigator user interfaces. This section is based on the assumption that you have already created, deployed, and tested your Datacap application in your development environment. If you have not done so yet, read the following chapters before you proceed:

- Chapter 6, “Structured forms application” on page 131
- Chapter 7, “Unstructured document application” on page 147

10.3.1 Overview

Configuring an application involves setting up jobs and tasks that run the user interfaces. Datacap Navigator includes administration features that allow you to create and edit the workflow, jobs, tasks, and shortcuts of an application. These features are available by selecting the Datacap Administration View as shown in Figure 10-9. You can configure any type of task, not just Navigator tasks, from the administration view.

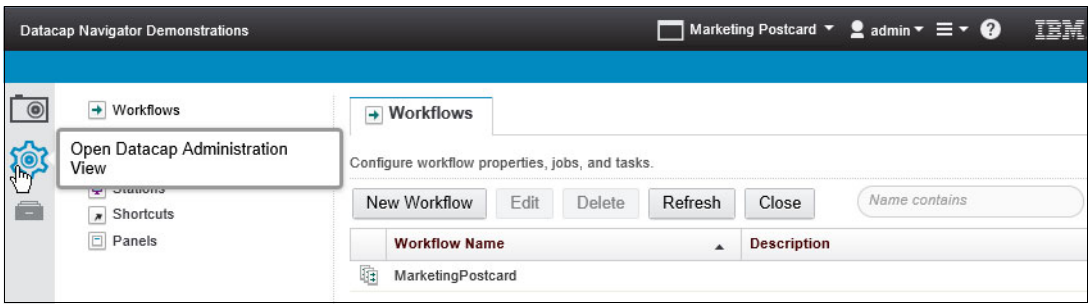


Figure 10-9 Opening the Administration View

The example we use is the Marketing Postcard application that is also used in the other chapters in this book. We need to add new jobs and tasks, and edit the shortcuts. The new jobs use the new Datacap Navigator user interface tasks. The jobs can use the existing background tasks, but when those tasks branch or split, they need to use the new Datacap Navigator jobs with the correct user interfaces.

Shortcuts are needed for the upload, verify, and fix-up tasks so that the tasks display for the user in the Shortcut panel. This is done by editing the existing shortcuts and selecting the new tasks.

10.3.2 Datacap Navigator job and task requirements

For this application, we need to add one job as the main job and additional jobs for verification and fix-up. Table 10-1 lists these jobs.

Table 10-1 Job names for Datacap Navigator

Job name	Description
Navigator Job	Main job for processing documents using the scan.js program and branching and splitting to Verify Export Navigator and Fixup Navigator as needed.
Verify Export Navigator	Process problem documents and enter data using the `verify.js` program.
Fixup Navigator	Process batches that have invalid structure using the `classify.js` program.

You can add the new jobs from the Administration view by selecting the workflow, clicking **Add Job** and adding the tasks to the jobs. Figure 10-10 shows the jobs.

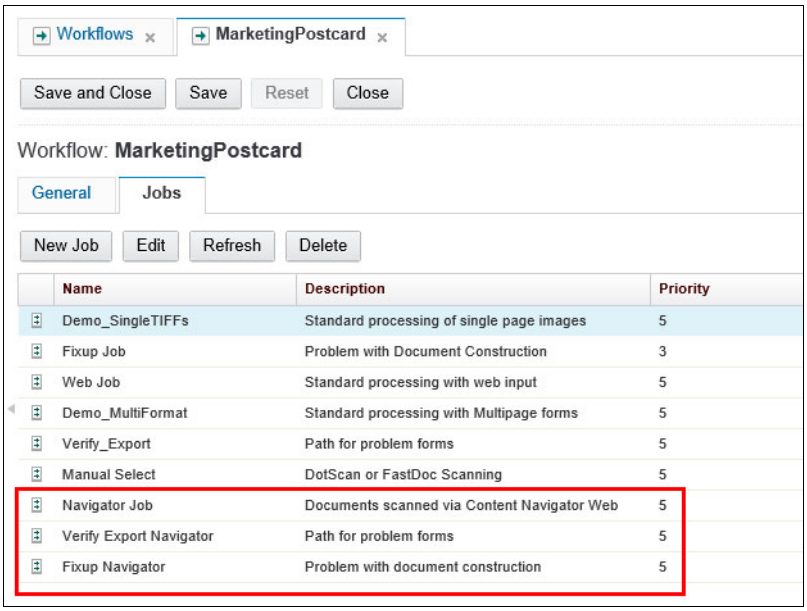


Figure 10-10 Jobs in the Administration view

New tasks are needed for the Navigator user-interface steps. Table 10-2 lists these tasks.

Table 10-2 Task names for Datacap Navigator

Task name	Program	Description
NScan	`scan.js`	Scan and import files
NUpload	`upload.js`	Upload scanned images and files to a batch folder
NFixup	`classify.js`	Manually classify and restructure a batch
NVerify	`verify.js`	Verify documents and data

You should add the Fixup Navigator and Verify Export Navigator jobs first so that they are available to be referenced by the main Navigator Job. Then, add the Navigator Job. The next sections describe the jobs in that order.

10.3.3 Fixup Navigator job

The Fixup Navigator Job (Figure 10-11) has only one step, which runs the NFixup task.

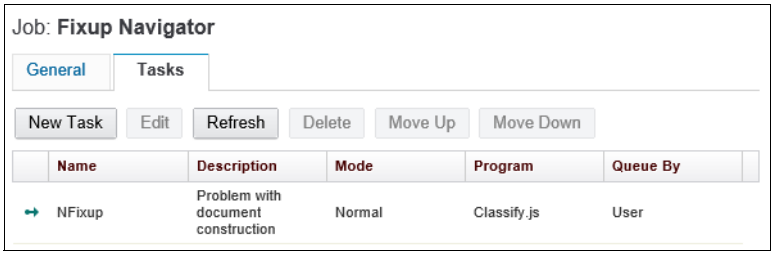


Figure 10-11 Fixup Navigator job

The NFixup task settings define the presentation and execution of the fix-up task in Datacap Navigator. The task needs to use the `classify.js` program as shown in Figure 10-12. The Advanced tab includes additional settings for configuring the `classify.js` user options. In our example we use the default values on the Advanced tab.

Figure 10-12 shows the NFixup user interface.

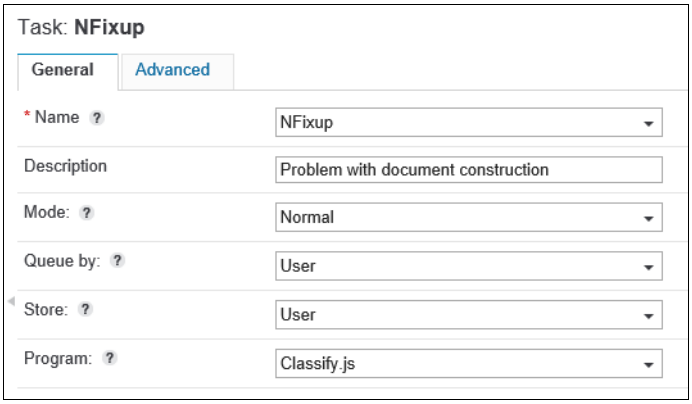


Figure 10-12 NFixup task settings

10.3.4 Verify Export Navigator job

The Verify Export Navigator job (Figure 10-13) includes a step that runs the Verify task in Datacap Navigator. Because the Export task does not need a user interface, it can be included in the job without modification.

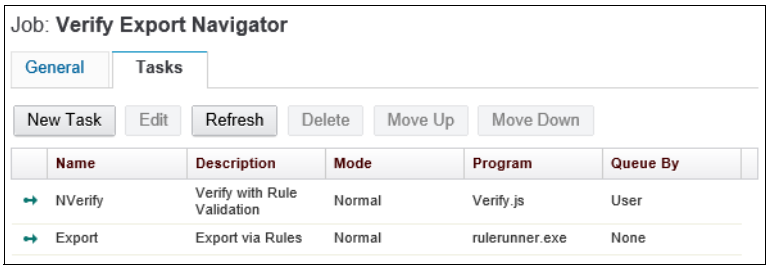


Figure 10-13 Verify Export navigator job

The NVerify task settings define the presentation and execution of the Verify task in Datacap Navigator. The task needs to use the `verify.js` program as shown in Figure 10-14. The Advanced tab includes additional settings for configuring the classify.js user options. In our example we use the default values on the Advanced tab.

Figure 10-14 shows the NVerify user interface.

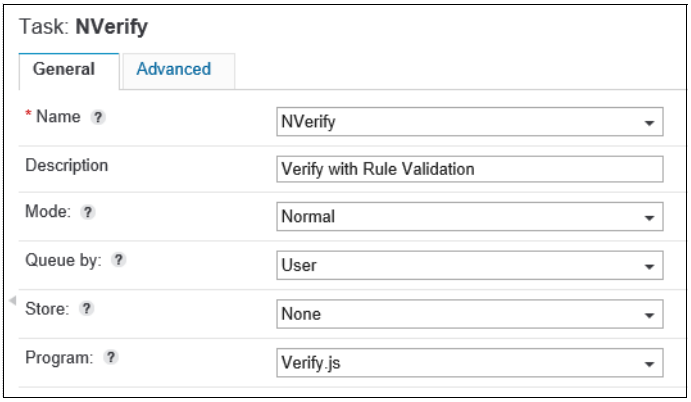


Figure 10-14 NVerify user interface

10.3.5 Navigator main job

The Navigator job can be defined after the Fix and Verify jobs are defined. It is the main job that performs the scanning and initial background task processing. It uses new tasks called NScan and NUpload. The background tasks PageID and Profiler are reused but must be configured to branch to the new Fixup Navigator and Verify Navigator jobs. Because the Export task does not need a user interface, it can be included in the job without modification. Figure 10-15 shows the tasks in this job.

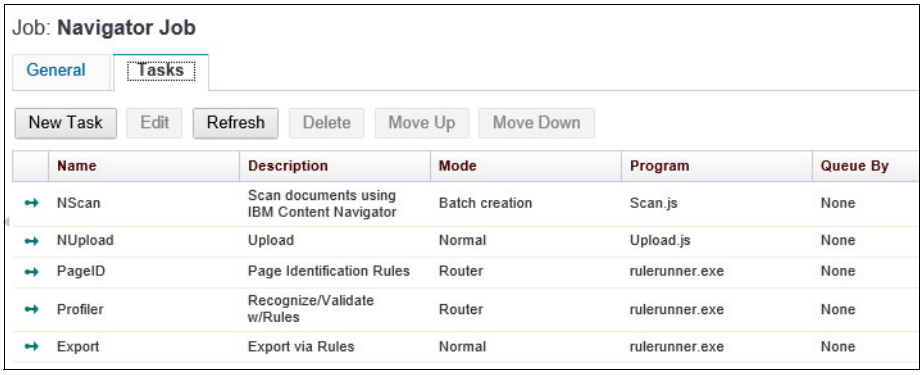


Figure 10-15 Navigator Job

NScan task

The first task in the job is the scanning task, NScan. The NScan task settings define the presentation and execution of the scanning task. The task must use the `Scan.js` program as shown in Figure 10-16. You cannot use the Multiple program setting with Datacap Navigator. The Advanced tab includes additional settings for configuring the scanning user interface.

The user's view of this task was shown previously in Figure 10-4 on page 237.

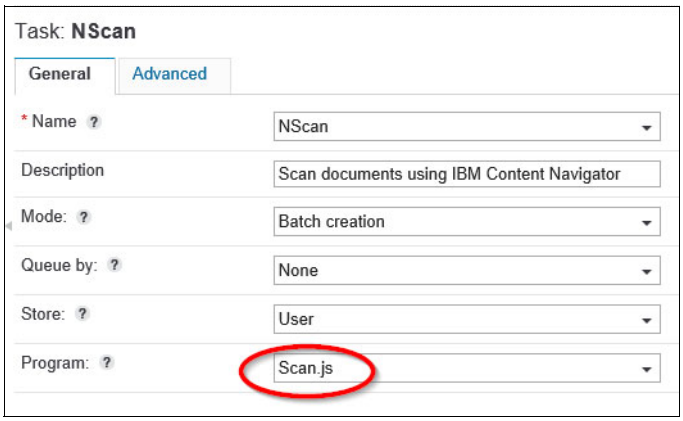


Figure 10-16 NScan user interface

NUpload task

The NUpload task is the second task in the job. Its settings define the presentation and execution of the upload task in Datacap Navigator. The task must use the `upload.js` program, as shown in Figure 10-17. The Advanced tab includes additional settings for configuring the upload user options.

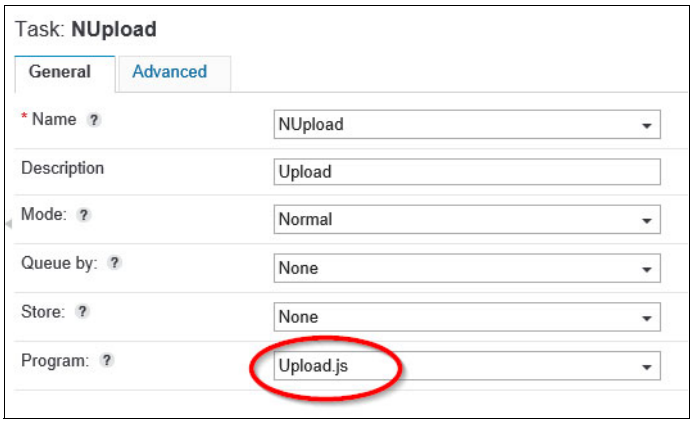


Figure 10-17 NUpload task settings

PageID task

The third task in the Navigator job is the PageID task. If the task detects an error condition, it branches to the fix-up user interface. This is configured to branch to a child job, Fixup Navigator (Figure 10-18). The Condition attributes must be set to implement the branch operation. The rest of the settings are defaults.

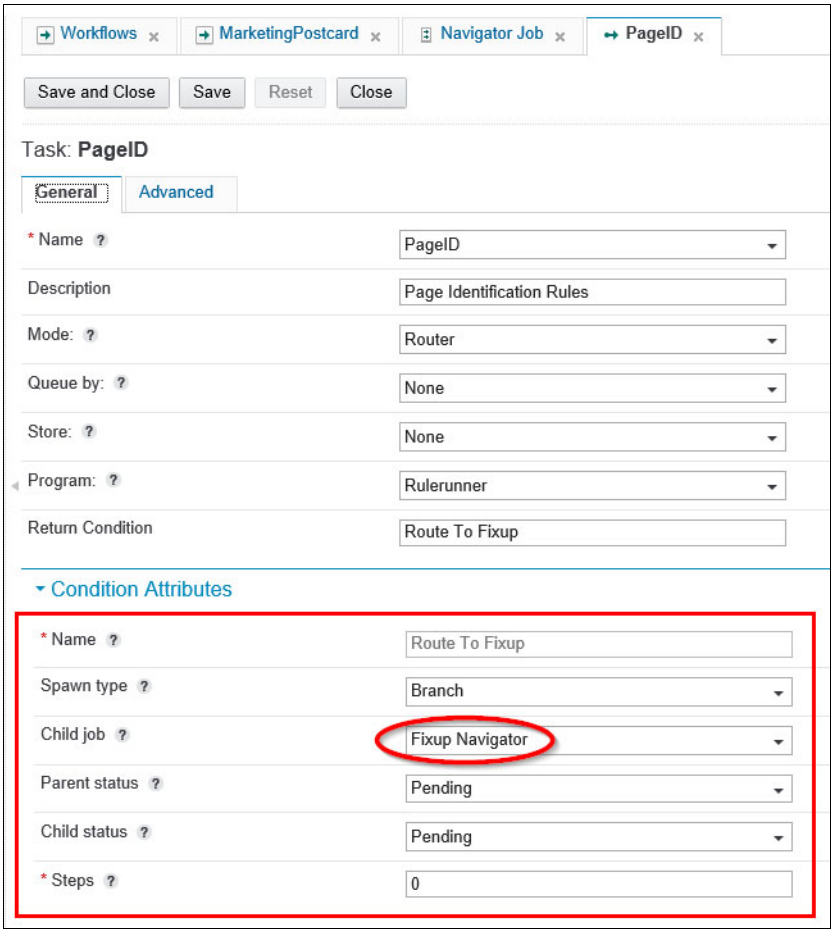


Figure 10-18 PageID settings for Datacap Navigator

Profiler task

The fourth task in the Navigator job is the Profiler task. If the task detects an error condition, it splits to the verification user interface. This is configured to split to a child job, Verify Export Navigator (Figure 10-19). The Condition attributes must be set to implement the split operation. The rest of the settings are defaults.

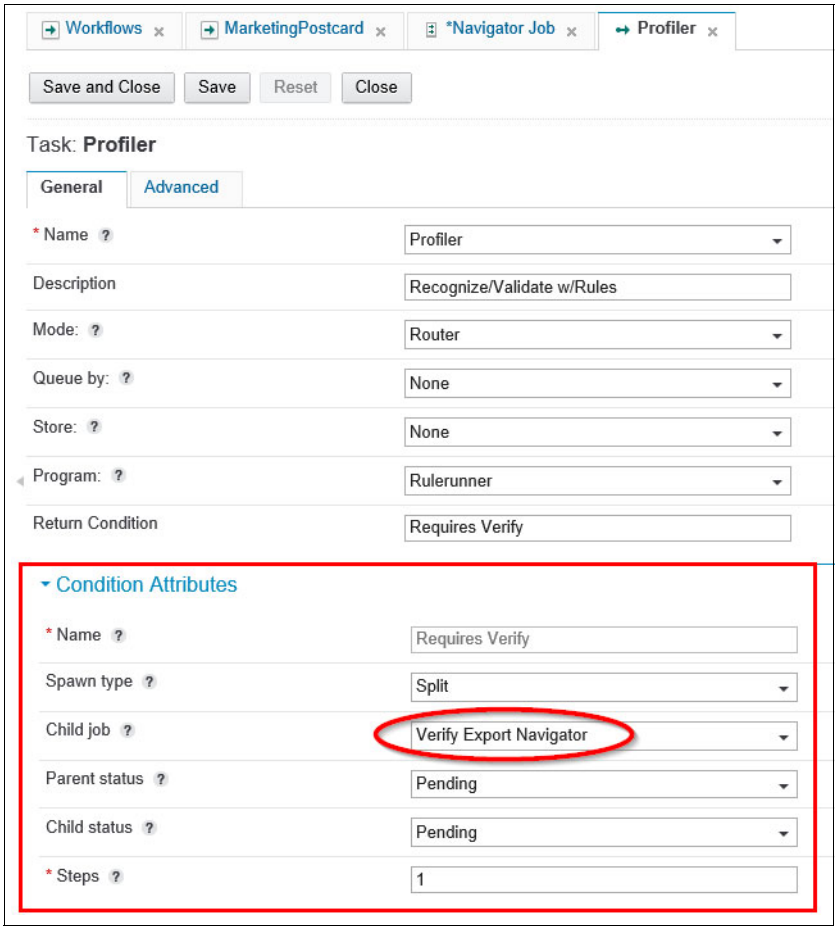


Figure 10-19 Profiler task settings for Datacap Navigator

One more update is needed. Shortcuts must be updated or defined to run the new tasks. These are created and edited in the Datacap Administration view as shown in Figure 10-20. Each of the user interface shortcuts needs to have the new tasks selected. If you want to run a background task in Datacap Desktop client, you can also select these tasks to the background task shortcuts.

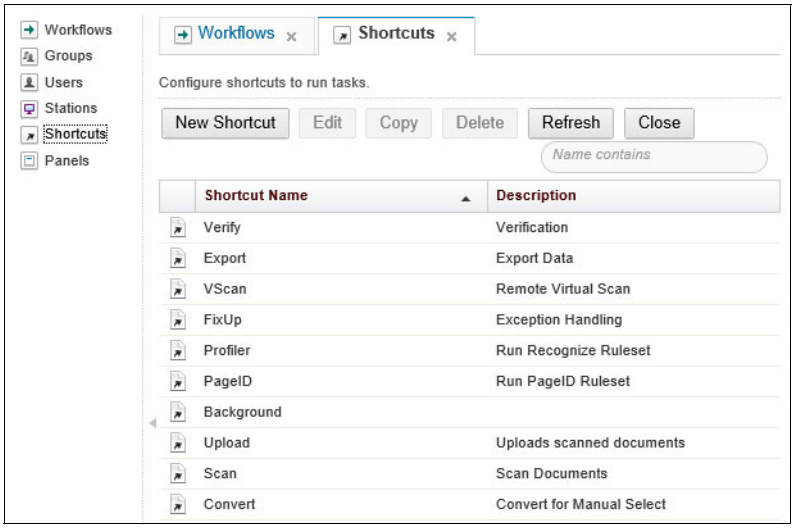


Figure 10-20 Shortcuts in Administration view

Figure 10-21 shows an example of the Scan shortcut configuration that has been updated to select the NScan task. Be sure to add shortcut selections for any Navigator job or task that you want to display on the Content Navigator user interface.

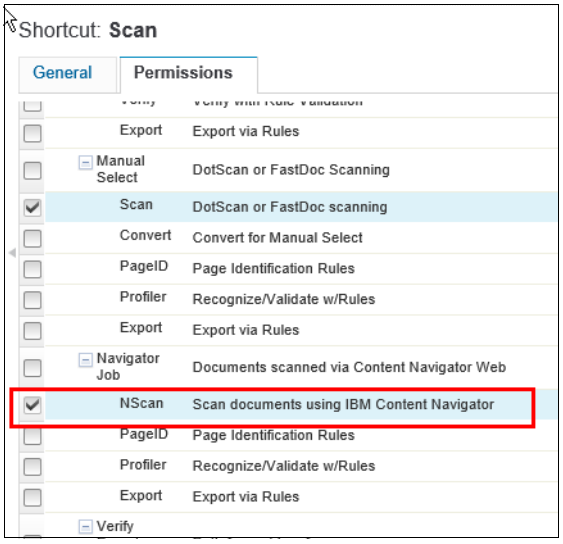


Figure 10-21 Scan shortcut example

With these configuration changes, the application can now run from Datacap Navigator. Each of the user interface steps use the new web interface.

For the background tasks run on the Datacap Rulerunner Server, you must add them to the Datacap Rulerunner configuration as described in 8.2, “Datacap Rulerunner” on page 190.

10.3.6 Defining custom panels

This section describes how to manipulate the custom panels in Panel Designer within the Datacap Administration View. One helpful aspect of this feature is that it uses the same technology as Case Manager, so if you are familiar with the Properties View Designer there, you will notice that the tool and techniques are similar.

Augmenting the system-generated panels is useful for many different scenarios. For example, the business analyst might want to include only a subset of fields, control the ordering of fields on the page, or place the fields into a layout which includes multiple columns and tabs. The analyst might also need to set up different views for different tasks.

When you verify a document, the system displays the fields on a Field Details panel. By default, the panel is dynamically generated by the system and requires no additional setup. The system-generated panel has a standard format which consists of all properties on a page arranged vertically in the order they are defined in the Datacap hierarchy (DCO).

Fields are also displayed in the Scan interface on the Start panel. When you scan a batch, the image viewer is displayed along with the associated panel defined in the scan task. This panel displays the batch level fields that apply to all of the pages in the batch. This panel is also dynamically generated by the system and requires no additional setup. The system-generated panel has a standard format which consists of all properties on a batch arranged vertically in the order they are defined in the DCO.

One more place where fields are displayed is on the Batch Editor within the Job Monitor. This panel is also dynamically generated by the system and requires no additional setup. The system-generated panel has a standard format which consists of all the Job Monitor and batch system properties. If a custom Batch Editor panel is created, there is no need to configure tasks settings, the system will fetch the last Batch Editor panel created for the application.

The Panel Designer can create any number of custom panels for each page type and batch type. It can create one custom panel for the Batch Editor. Before accessing the Panel Designer, ensure that the wanted fields are defined for a page type in FastDoc (admin) or Datacap Studio.

10.3.7 Custom Verify task panel

A Verify panel is associated with a Datacap page type on a Verify task. So when a specify page type is selected, the associated Verify panel defined for the page is displayed. The panel defines the type of layout and Datacap fields to be displayed, and for each field, settings such as label, help hint, required or read-only, default value, masking, length, and so on.

A page type can have more than one panel. In this case, you could have different panels for different Verify tasks. For example, you could have two Verify tasks where the first task enters and corrects the data and the second task verifies the data in a second pass by a different user.

Follow these steps to create a panel:

1. Select **Panels** on the administration list on the left side of the window as shown in Figure 10-22.

   This lists any custom panels that had already been created for your application and allow you to create new panels. The first time that you select it, the list is empty.

2. Click **New Panel** and select **Verification Panel** from the drop-down list.

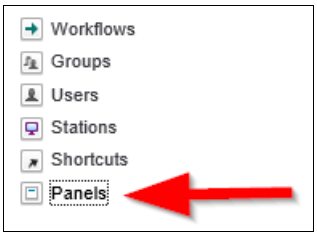


Figure 10-22 Selecting Panels to create a panel

3. From the Page Type drop-down list, select the page type for your panel. A default panel layout is constructed in the center. By default, all fields are listed vertically.

Figure 10-23 shows a custom panel that is designed for the Marketing Postcard application.

The upper left pane has the different containers to drag onto the canvas, which is the center of the Panel Designer in the middle pane. The lower left pane has the fields of the page type to drag into the different containers on the canvas. Initially, the canvas is populated with all of the fields in a single column. You can rearrange the fields by dragging them around the canvas.

When you select a field on the canvas, the settings for the field display on the right panel. This enables you to configure the individual field settings and tailor the runtime behavior at the field level as the user moves from field to field.

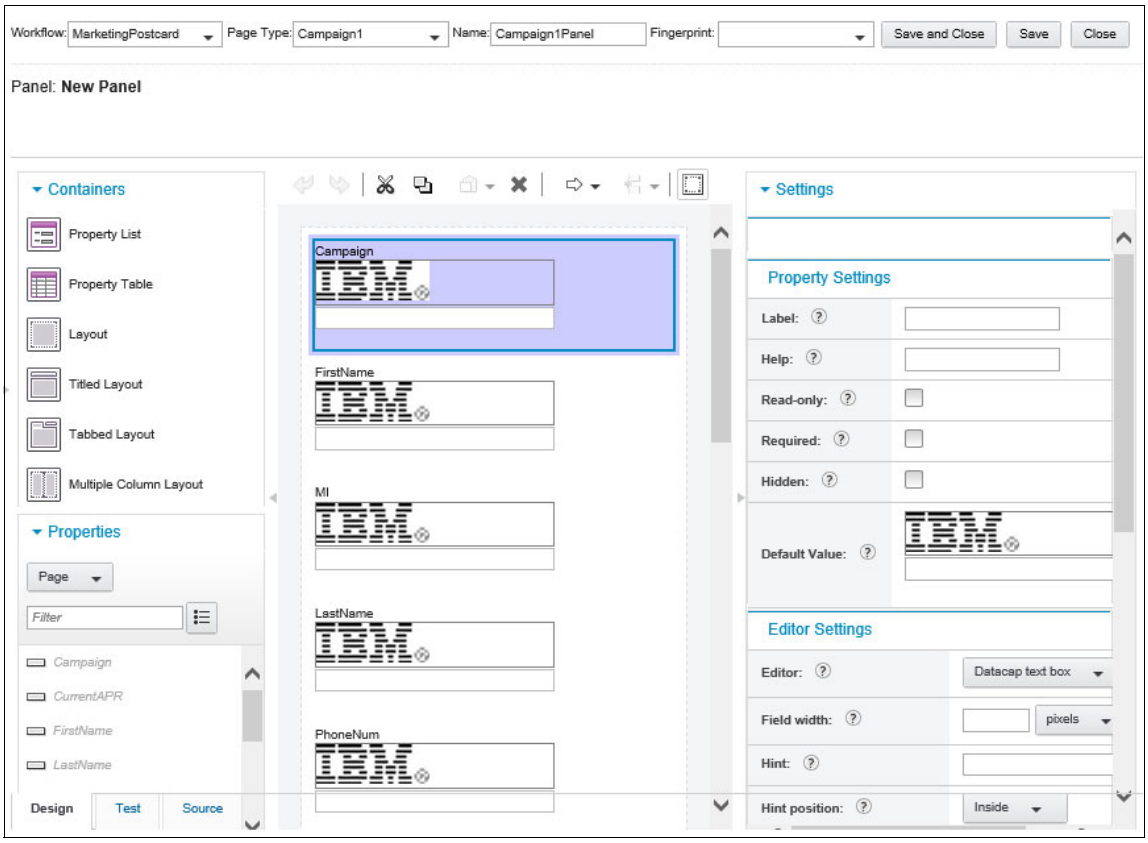


Figure 10-23 Custom panel

4. To deploy the panel, configure the NVerify task settings as shown in Figure 10-24 on page 253. You add the custom panel to a list of panels that map the page types to the panel names. Within the NVerify task settings, select the **Advanced** tab and scroll down to the Custom web panels section. Select the **Use custom web panels?** and complete the page type and panel name.

If a page type is not included in the list, the system generates a panel. Therefore, a custom panel is not needed for every page type.

In Figure 10-24, the custom panel is used in the NVerify user interface.

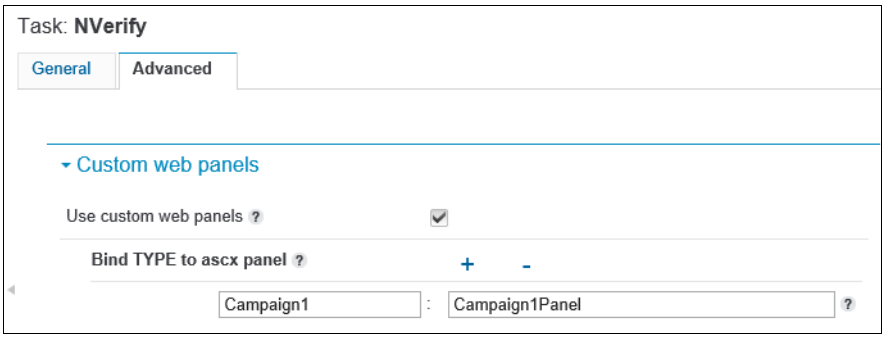


Figure 10-24 Custom web panels

10.3.8 External Data Services

Datacap Navigator includes a method called External Data Services to access data an external source such as a file or a table in a database, to customize field properties, and to manage property behavior in the user interface. You access the data without moving or copying that data to a separate repository, so the source remains in the original data store. The external data source must remain available to IBM Content Navigator so that the external data can be accessed whenever business users invoke the service through the web client.

You can use an external data service to customize the field properties and property behaviors that the subsections that follow describe.

Look up values in a database to create choice lists

Create choice lists by using existing data that is managed in a different content repository or data source than the one that is connected to IBM Content Navigator.

For example, you can use values in a file that is located and managed in an external server or repository.

Prefill properties

Specify prefilled properties and default values.

For example, you can prefill fields with custom default values that are based on a particular class ID, authenticated user, or the parent folder.

Specify property dependencies

Define dependencies between properties.

For example, you might specify a dependency between a geographic region choice list property and an office branch choice list property so that when a user chooses a geographic region, the subsequent choice list that is dependent on the selected geographic region contains only the office branches that pertain to that geographic region.

Set minimum and maximum values

Specify an integer, float, or date to define the maximum or minimum value for a property.

You cannot reset the minimum value or maximum value to be less restrictive than the minimum value or maximum value that is specified in the repository that you are using. For example, if the minimum value in the repository is 100, the service can set the value to 150, but not to 50.

Set read-only status

Set a property to be a read-only field. For example, you might create a property that requires a particular value. To prevent users from entering a different value that could cause an error, you can specify the correct default value and make that property read-only.

Set required status

Set a property to be a required field. When you use this attribute on a property, an asterisk appears in the user interface to indicate that the field is required. Users cannot proceed from the page or dialog box unless the field contains a value.

Set hidden status

For example, you might create a choice list that dynamically determines subsequent text input fields to present in a form. To hide a property that does not apply in a particular situation, you can use the hidden attribute.

Implement property validation and error checking

Show a custom message or provide assistance when users enter values into a property field.

When an external data service is implemented for a certain action or property, the service is invoked when a business user interacts with that item in the web client.

Figure 10-25 shows how an external data service submits and returns requests.

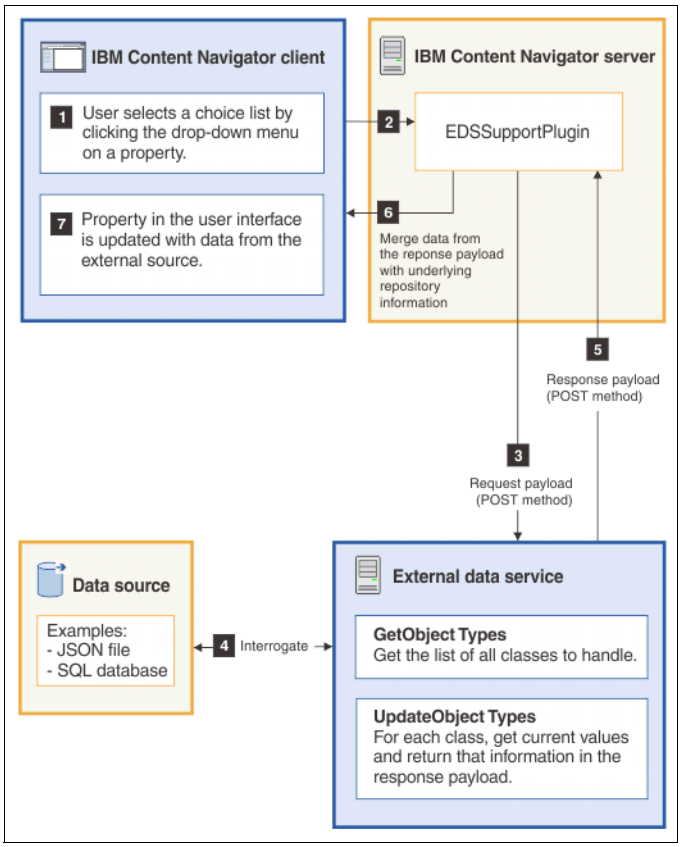


Figure 10-25 Process of external data service

For more information about External Data Services, see the IBM Redbooks publication titled *Customizing and Extending IBM Content Navigator*, SG24-8055 and the Content Navigator documention in IBM Knowledge Center:

http://ibm.co/1J058vv

11

Datacap Mobile user experience

This chapter provides information about the capabilities and considerations of using readily available mobile devices such as mobile phones or tablet devices for document capture.

IBM provides a Datacap Mobile app that is available on both Android and iOS platforms, allowing direct capture of images from a mobile device with an embedded camera at the point of origination. The app supports on-device document and page classification and optical character recognition (OCR), allowing the user to quickly and accurately scan, categorize, and index one or more documents for submission into a back-end system, such as an enterprise content management repository. The app supports a flexible deployment methodology and can be further customized, by the customer or a Business Partner, using a Software Development Kit (SDK).

The app is built to connect to a Datacap server, so it supports any Datacap version 9 application that has been configured for mobile capture.

This chapter includes the following sections:

- Overview
- Typical mobile capture use cases
- Mobile capture app configuration
- Capturing using Datacap Mobile
- Viewing captured content
- Deploying Datacap Mobile
- Representational state transfer (REST)

11.1 Overview

In today's market, customers expect answers to their questions in real time. They expect paper to be processed in a similar time frame as digital requests. In short, the customer expects more and faster service.

One way to meet this demand is to capture paper-based documents at the point of origin; that is, capturing the image while the signature is still drying. By capturing immediately, delays in transferring paper to a central location for preparation and scanning are removed or eliminated. It also removes the risk of loss or theft of highly sensitive documents. Furthermore, allowing indexing of documents at the point of origin ensures a greater degree of completeness and accuracy, reducing the risk of further delay while incomplete or inaccurate information is identified and corrected.

Other use case scenarios include mobile workers who are away from the office for extended periods of time. They might also work in remote locations where transport of a dedicated scanning device is simply not practical.

Historically, images were captured solely on dedicated scanning devices using either a sheet feeder, or a glass panel. These devices are typically set up and maintained for optimum scanning of documents. The rise in use of the smart mobile device now allows documents to be captured through another medium, the mobile camera.

Customers want to capture their documents, but they also want to view the documents after the documents have been captured and stored. Enabling this capability on a mobile device allows easy access from any location with a suitable network connection (Figure 11-1). No longer is a desktop computer the only device that can be used to access your documents.

Figure 11-1 Capture is one of many uses for a smart mobile device

11.1.1 Considerations for image capture

Use of a camera on a mobile device differs significantly from the traditional scanning approach. Cameras can distort images in a variety of ways. In the following sections, we cover a few of the challenges that have to be considered when using a mobile device.

Exposure

Exposure is about how much light is let into the camera. Too much light, and your images appear washed out; too little light, and they will be too dark. The automatic capture mode in Datacap Mobile handles the exposure and enhances the image to optimize the contrast, as you will see in the use case example. The app also provides manual adjustments that can also correct a poorly displayed image, but only to a certain degree.

An example of good, under-exposed and over-exposed, is shown in Figure 11-2.

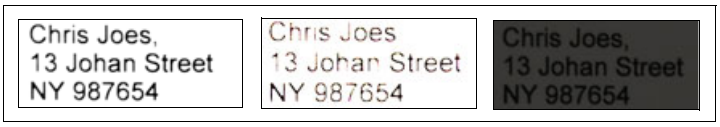


Figure 11-2 Examples of good, over-exposed, and under-exposed images

Angle of capture

If the paper is not aligned to the camera lens, it can appear distorted. This is known as the *keystone* or *tombstone* effect. The captured image results in dimension distortion, making it look like a trapezoid, the shape of an architectural keystone. When applying technologies such as optical character recognition (OCR), this can cause undesirable results due to differences in character size and shape.

It also does not provide a suitable representation of the original document for storage. The Datacap Mobile app automatically handles this concern by automatically straightening the image. Examples of an angled image and a fixed image are shown in Figure 11-3.

Figure 11-3 Examples of an angled image and fixed image

Image blur

A mobile device is operated by a user holding the device in their hand, so there is the possibility of the user's hand shaking or wobbling as the image is captured. This can cause image blurring not usually encountered when using a stable desk scanner. In turn, this can affect the quality of the captured image, affecting the definition of the image characters and therefore the OCR results.

In Datacap Mobile, the software examines the video stream from the camera and only takes a photo when the image is in focus which minimizes blurred images. An example of a blurred and fixed image is shown in Figure 11-4.

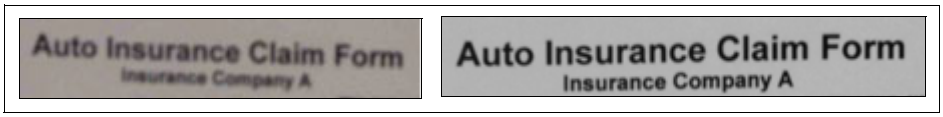


Figure 11-4 Example of a blurred and fixed image

Skewing

Although skewing still occurs on dedicated scanners, it can be more acute on a mobile device. When using a mobile camera, there are no physical guidelines or rails to align images against. Therefore, obtaining that perfectly aligned scan is tricky. Datacap Mobile handles this concern by automatically straightening the image.

An example of a skewed and deskewed image is shown in Figure 11-5.

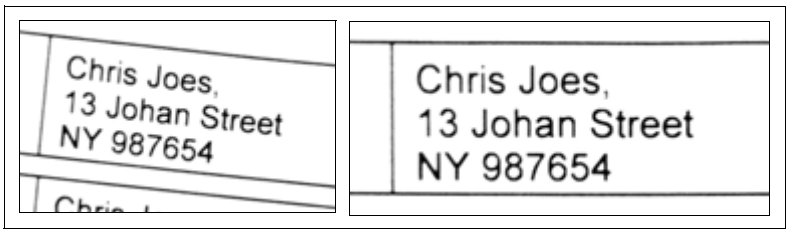


Figure 11-5 Example of a skewed and deskewed image

Resolution

The image resolution used determines the level of detail used to capture an image. The greater the resolution in dots per inch (dpi), the more detail that can be captured, and therefore interrogated, during the capture process. Similarly, the lower the resolution used, the less detail that can be captured. The result is that OCR results can differ greatly depending on the resolution used.

On a mobile device, the focal distance from the physical object is typically unknown and, as a result, the resolution is also unknown. However, it can be calculated approximately and assigned before running the OCR, based on the known physical size of the document and the definition (x pixels by y pixels) of the camera and how much of the document was captured in the camera window.

The issue here is that the higher the resolution used, the larger the file size, therefore increasing storage costs. In the case of mobile capture, the amount of bandwidth required to transfer the image also increases with image size, which can also exacerbate cost and upload times. An optimum image size and resolution are therefore desirable. An example of a high-resolution and low-resolution image is shown in Figure 11-6.

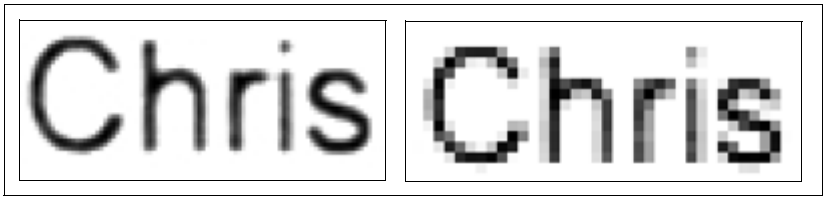


Figure 11-6 Example of a high-resolution and low-resolution image

Color versus black and white

Images can be captured in color or black and white. The choice depends on your application requirements. Consider file size here as well, color images require larger file size compared to black and white.

The user

With all of these issues, you must also take one other factor into consideration: The user. Mobile capture users are typically users who have had no prior education about how to prepare or scan documents optimally. Some will not be too concerned if the image is skewed or distorted. In their minds, the image is captured, so their job is complete. Therefore, you should look, where possible, to guide these users into capturing an optimal image.

The device

Different mobile device manufactures use different manufacturing processes to produce the camera in their products. This leads to differences in the image produced even though the source from different manufacturers might produce same image format, such as JPEG. Even within the 'same' device from a given manufacturer there are multiple camera resolution and definition options that can lead to different results.

For more information

All of the considerations described mean that the image produced can vary from device to device, from location to location, and from environment to environment. To achieve the best possible capture results, Datacap Mobile app is preconfigured to programmatically ensure the best possible image quality the moment the document is captured. When set to auto-capture, the app provides real-time detection of edges, deskews the image, and only snaps the image when the correct quality threshold is met. See 11.4.1, "Capturing in Auto mode" on page 264 for more information.

In cases where the app cannot obtain the best results automatically, the Datacap Mobile app provides additional capabilities to help the snap the best possible quality image manually, such as enabling flash for conditions of poor light and enabling capture in black and white. See 11.4.2, "Capturing in Manual mode" on page 267.

Also, the most important features of Datacap Mobile can be further extended and customized using the SDK. See 11.6.2, "Software development kit" on page 272 for additional details.

11.2 Typical mobile capture use cases

This section describes two possible use cases where mobile can be used to address real-life capture requirements.

11.2.1 Mobile banking

Customers using banks, and post offices in some countries, have requirements to transfer funds between their accounts and to other individuals. Historically, this was done by completing a form with the relevant information and handing it in at a branch or local post office. The form was then manually processed and scanned for storage. This process involved several manual steps and also required that the customer visit the branch in person within certain hours of the day, potentially wait in a queue to see a clerk, and then wait for the form to be processed. All of these steps added delay and effort to the process.

A possible way of expediting this process is by empowering the bank customers to capture the data. The widespread and ubiquitous nature of mobile phones and tablets around the world makes them an attractive choice of device to deliver this capability to a wider audience. The mobility and simplicity of mobile devices offer are also a plus.

A new and increasingly popular use case is onboarding customers for new services, such as a new credit card, where you need to provide IDs and proof of residency. In many cases, imaging is just one aspect of a larger customer-facing app, such as a banking app with many use cases, which is why offer the Datacap Mobile SDK. See section 11.6.2, “Software development kit” on page 272 for more information.

The end result is more empowered customers who are more in control of their money. Less personal time is wasted and the process is delivered faster.

11.2.2 Remote workers

Company employees who work out in the field might work using paper-based processes. These paper forms might require manual signing or completion and simply cannot be easily replaced by use of a direct digital process. For example, wet signatures might be required, or specific stamps might be needed to complete process. It might also be cost prohibitive to integrate automated processes between companies.

A possible way of addressing this is to use of mobile devices such as tablets or mobile phones. These devices can be centrally controlled, provide high-speed mobile network connectivity, are small, compact and intuitive to use.

Use of such devices allow workers to capture the paper forms at the time of completion and upload directly to a central automated process. No longer does the worker need to store physical copies until they return to a suitable physical location to either scan or post.

11.3 Mobile capture app configuration

This section details how to download and set up the IBM Datacap Mobile app for use on your device. It assumes that you already have a working installation of IBM Datacap 9 with network connectivity established between mobile device and Datacap server. It also assumes that you have a Datacap application that has been configured for Mobile capture. In our configuration example, we added a mobile job to our Datacap application.

11.3.1 Configuring Datacap Mobile

The following steps detail how to obtain the IBM Datacap Mobile app and configure for it use with your IBM Datacap system:

1. Download the IBM Datacap Mobile mobile app from either the Apple AppStore or Google Play.
2. After obtained the mobile app, open the app by tapping the **Datacap** icon on the device to launch the application.
3. If this is the first time running the app, you will be asked to configure the Connection properties, including the URL of the server running Datacap Web Services; the Datacap application you want to use; the capture workflow; and your login credentials.

4. Enter the URL of the Datacap Web Services, as shown in Figure 11-7.

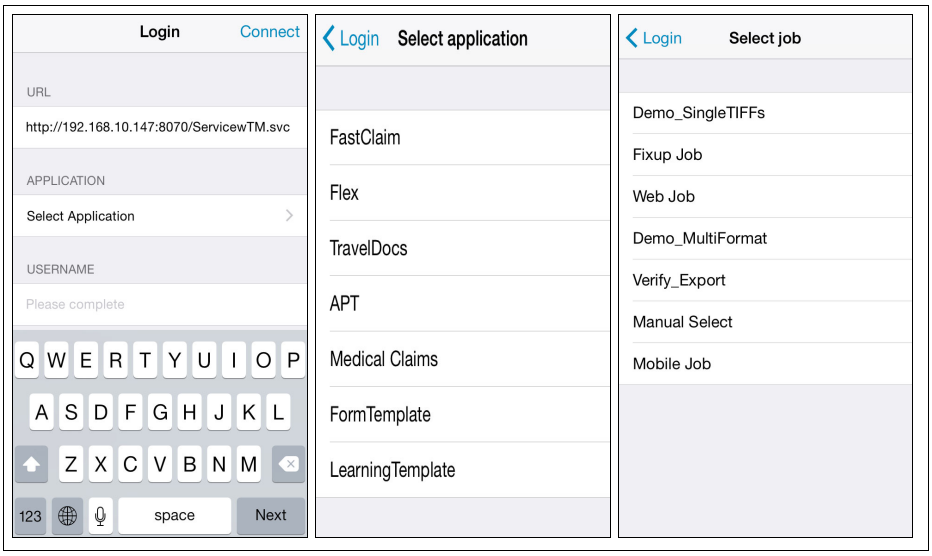


Figure 11-7 Configuring Datacap Mobile

5. Click **Select Application** and select the Datacap application that you want to use, as shown in Figure 11-7. In our example, we use **FastClaim**.
6. Enter your username, password, and station ID in the appropriate fields, and click **Connect**.
7. The final step of the configuration process is to select the appropriate mobile job or workflow. In our example, we select **Mobile Job**, also shown in Figure 11-7.

11.3.2 Server-side configuration

To use Datacap Mobile, you must install and configure the Datacap Web Service. Instructions are provided in “Datacap Web Services hosting options” in the IBM Knowledge Center:

http://ibm.co/1L2xtU5

Next, ensure the workflow or job you intend to use has a batch-creation task configured for mobile capture, as shown in Figure 11-8.

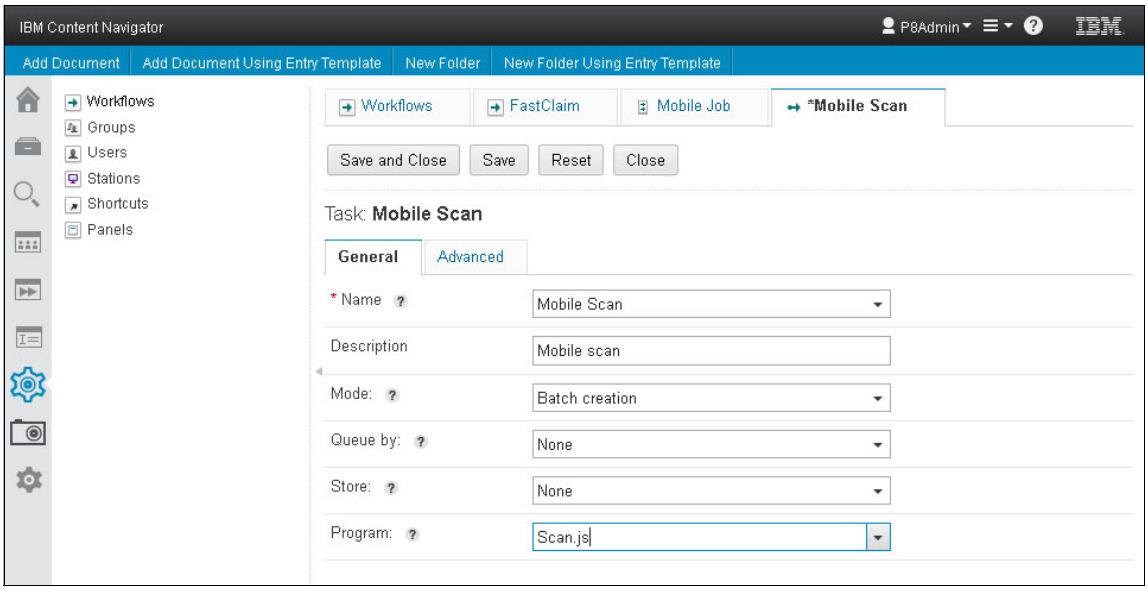


Figure 11-8 Workflow configuration with Datacap Navigator to enable mobile capture

11.4 Capturing using Datacap Mobile

Datacap Mobile brings advanced capture capabilities to the point of capture, improving the accuracy, speed, and reliability of a capture process. It does this by providing users with an intuitive user interface that allows them to quickly capture, classify, index and submit one of more documents for processing. Datacap Mobile supports these key functions on the mobile device:

- Capturing in Auto mode
- Capturing in Manual mode
- Classification
- Indexing

Additionally, Datacap Mobile provides several additional capabilities, such as offline support, which will be covered at the end of this section.

In this section, we walk through a capture process using Datacap Mobile.

11.4.1 Capturing in Auto mode

As described in 11.1.1, “Considerations for image capture” on page 259, there are many factors that affect the quality, and thus the expected OCR results, of documents captured using a mobile device. In order to reduce the learning curve for the user and to ensure the user gets the best possible results, Datacap Mobile is able to capture documents automatically, in Auto mode. This method of capture is the default configuration for Datacap Mobile, as shown in Figure 11-9 on page 265.

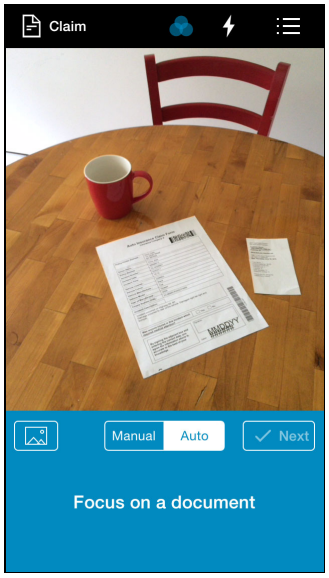


Figure 11-9 Auto mode is default capture method

With automatic capture, Datacap Mobile detects edges, deskews, and verifies ambient lighting is adequate before taking a snapshot. If it determines that quality levels are not sufficient, for example because of poor ambient lighting or because it has difficulty detecting edges against a background whose color is too close to that of the document that is contrast, it does not take the snapshot. In such cases, the user can make an additional attempt to have Datacap auto-capture the document using the device's light., by clicking the lightening icon in the menu bar at the top, or by physically moving the page to an area with better background contrast.

In our example, we need to capture several different documents and additional images:

- An auto claim page, containing details about the claim
- A supporting receipt
- An image of property damage

Together, these images comprise a single *claim* document.

The first image we want to capture is the main part of the Claim that we want to submit. It is the claim form containing demographic details about the customer, such as their name and contact information, and s details about the vehicle, a car in this case, that has been involved in an accident. We must ensure this image is clear, clean, and readable by an OCR engine. We must also ensure the image is legible for human readers as it might need to be preserved as a record for a given period of time. Datacap Mobile's Auto capture ensures that you capture the best possible image.

To start the capture process, we use Datacap Mobile's Auto capture capability simply by holding our mobile device over the page and ensuring that Datacap Mobile is able to "see" the page in its entirety.

Datacap Mobile auto-detects the edges of the page, as shown in Figure 11-10.

Figure 11-10 Capturing an image in Auto mode

If the image is of sufficient quality, Datacap Mobile snaps a picture and displays a thumbnail of the image in the bottom quarter of the window, as shown in Figure 11-11.

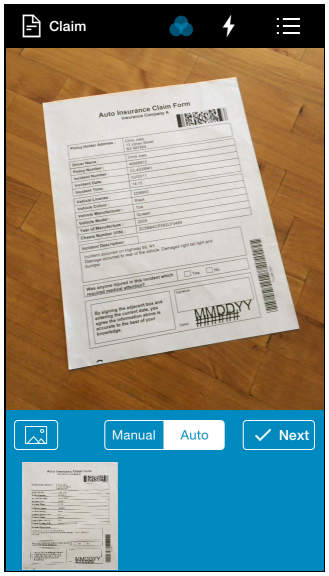


Figure 11-11 Captured page thumbnail displayed is displayed at the bottom of the window

At this point, the user might want to capture additional pages automatically or, depending on the use of the image, switch to **Manual** mode for additional capture.

11.4.2 Capturing in Manual mode

Some images, such as photographic evidence of property damage, are not suitable for automatic capture because there is no notion of a “page” with edges, minimum levels of quality for OCR, and so on. Even some content, such as a receipt, might not need to be read by machine and instead might just need to be submitted as evidence along with a main document, as is the case in our example. Such images can be captured by using Datacap Mobile in Manual mode, as show in Figure 11-12.

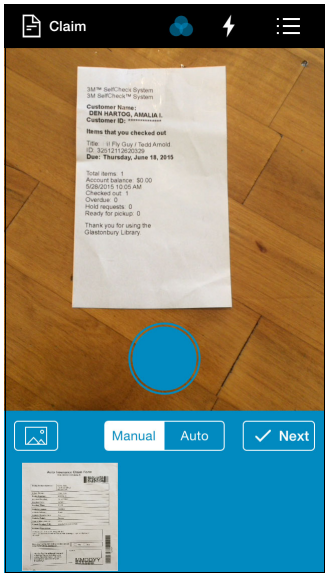


Figure 11-12 Datacap Mobile in Manual mode

In Manual mode, Datacap Mobile behaves like the on-device camera familiar to mobile users. It allows the user to point-and-click and snap pictures of anything that is relevant to the document being submitted. Just like in Auto mode, each individual image is displayed as a thumbnail icon at the bottom of the window.

A third method of adding images to the document is by clicking the Album icon, shown in Figure 11-13. This opens the default photo album on the device and allows the user to select and add images to the captured document.

Figure 11-13 Album icon

Note: The methods of adding documents can be used in any order, as suits the user’s situation and use case. Datacap Mobile is a versatile tool that fits any real-world capture use case.

In our example, when the user has captured the Claim form, manually captured a receipt using Datacap Mobile, and added a third image from their photo album, the user sees thumbnail icons of all images, shown in Figure 11-14, at the bottom of the window.

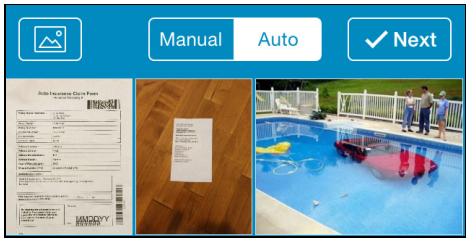


Figure 11-14 Thumbnails of all images captured

The user clicks **Next** to move to the Classification page.

11.4.3 Classification

Classification, or identification, is the process of determining what pages we are dealing with and how they fit together into a document. Datacap applications are generally configured to classify pages, assemble them into documents, and then extract machine or hand printed data from the documents, in that order. In Datacap Mobile, we can use the user to help Datacap accurately and correctly classify pages and documents, ensuring greater accuracy and fewer errors “downstream”.

In Datacap Mobile, there are two methods for classification: the user selects the document type up front (or accepts the default selection) and then identifies individual pages within the document or, the user snaps images first and later adds them to a document and identifies the pages.

In our example Datacap application, there are two document types, `Claim`, which is selected by default, and `Expenses`, as shown in Figure 11-15. All the images we captured initially are automatically a part of the Claim document.

Figure 11-15 Select document type

By clicking **Next**, the user proceeds to classify each page within the Claim document, as shown in Figure 11-16 on page 269. The user can tap Edit and move or delete pages. The user can also drag pages into other documents that the application supports, such as Expenses, in this example.

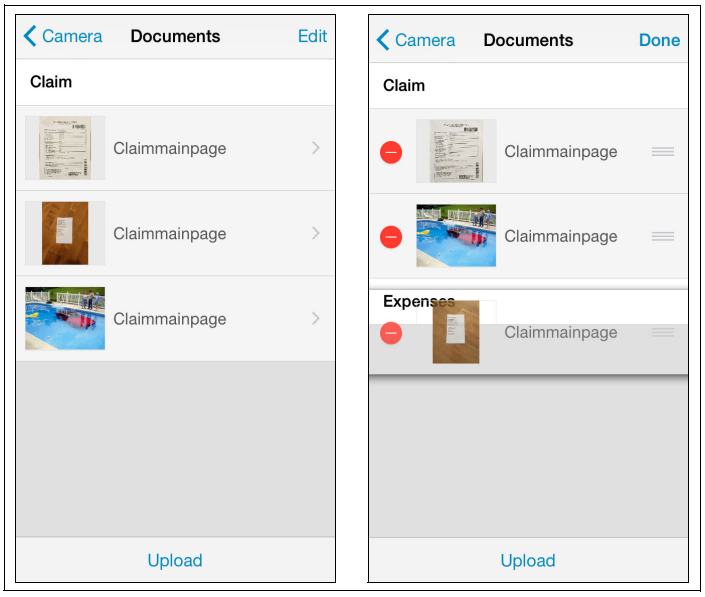


Figure 11-16 Document editing

The user can also tap each individual page and identify it correctly, based on the page definitions in the Datacap Application, as shown in Figure 11-17.

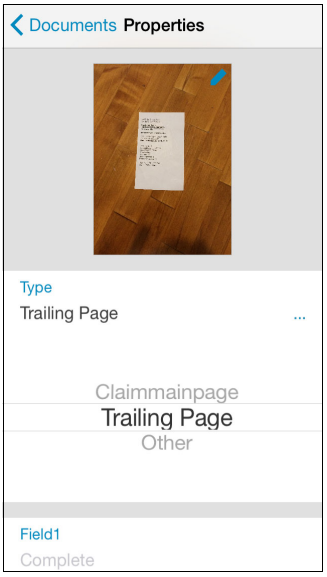


Figure 11-17 Page identification

11.4.4 Indexing

The final step in preparing the Claim document for submission to the Datacap server is to extract data from the images into designated fields. In our example, the fields include values such as the name of the driver, the policy number, and the date the incident took place. Although indexing can be done server-side, Datacap Mobile allows us, again, to use the user to ensure indexing is done accurately and comprehensively, ensuring fewer errors later in the business process. With accurate indexing at the point of capture, we can extract value from documents sooner and respond to the customer's needs faster.

Datacap Mobile supports three types of indexing:

- Manual
- On-device OCR
- Bar code recognition

Manual

Manual indexing is the “traditional” method of entering data on a mobile device: the user simply types the values into the field. This process is slow and might lead to errors.

On-device OCR

On-device optical character recognition allows the use to identify and select the data they want to extract. Datacap Mobile then extracts the value using OCR and populates the field.

To extract data using OCR, the user taps the field they want to enter data into and clicks the **Read Text** option at the bottom of the window. The user then pinches and zooms to the desired location on the page and taps the **Select** icon, as shown in Figure 11-18.

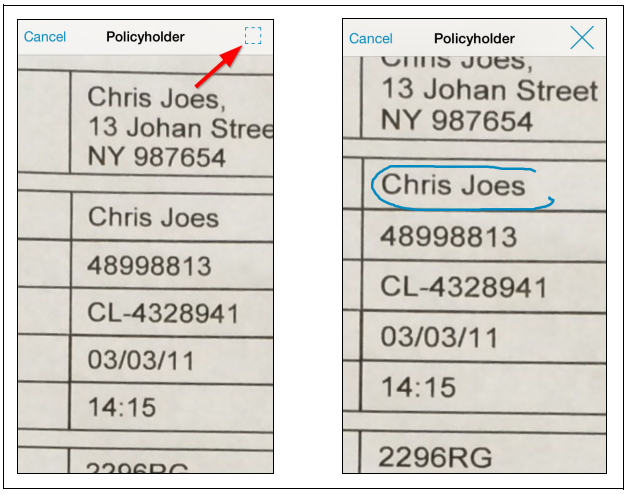


Figure 11-18 On-device OCR: Click icon and select text

Next, the user circles the text they want to extract, also shown in Figure 11-18.

The selected data is then added to the field, *Policyholder* in this example. The user can quickly verify that there were no OCR errors and move on to the next field.

Bar code recognition

Bar code recognition requires that the user snap a close-up of a given bar code on the document. To get started, the user taps a field, as normal, but rather than selecting Read Text they tap **Read bar code**. They then focus the camera of the mobile device on the bar code they want to capture, as show in Figure 11-19 on page 271. When captured, the extracted value is added to the field.

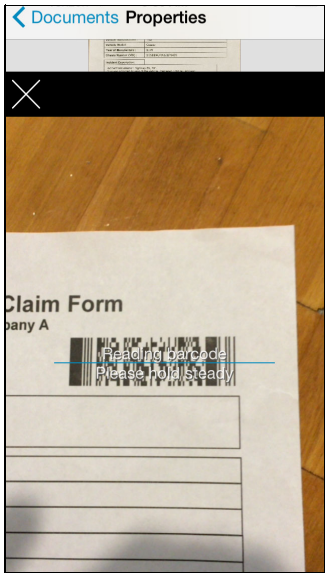


Figure 11-19 Capturing a bar code

11.4.5 Submission

With capture, classification, and indexing complete we can now submit the document to the Datacap server for further processing. The user does this by returning to the Documents window and tapping **Upload**. Because Datacap Mobile supports capturing documents in offline mode, the newly submitted document is queued for upload to the server as a background process. The user can see what documents are queued for upload by tapping the settings icon in the top right corner of the window.

11.5 Viewing captured content

Content captured using Datacap Mobile can be accessed and viewed in several ways, depending on the business need, the access rights that have been granted to a given user, and the location of the user. On a desktop computer or a notebook, the user might access the content through IBM Content Navigator or through a solution built on top of Content Navigator, such as IBM Case Manager. For example, in our Claims scenario, filing the claim document through Datacap Mobile might automatically create a case in Case Manager, with the case folder contain the Claim form and the supporting documents that were captured using the mobile device.

If the user is on a mobile device, she can access captured content using a tool such as the IBM Navigator or Case Manager mobile app.

11.6 Deploying Datacap Mobile

Datacap Mobile can be deployed in several ways, depending on the business need. The deployment options available are described in this section. They include:

- Unmodified (‘shrink-wrapped’)
- Software development kit

Each option is described in this section.

11.6.1 Unmodified (‘shrink-wrapped’)

The simplest deployment method is to download and use Datacap Mobile directly from the Apple Store or Google Play without any further modifications beyond the setup and configuration described in 11.3.1, “Configuring Datacap Mobile” on page 262. This method, also referred to as using the “shrink-wrapped” version of the app, requires nothing more than configuring your Datacap applications for mobile access; configuring network and firewall access; and setting up Datacap Mobile.

11.6.2 Software development kit

The most comprehensive method of deploying and customizing Datacap Mobile is to use the Datacap Mobile Software Development Kit (SDK). This gives you complete freedom to integrate Datacap Mobile’s capture function into your own application, including accessing features and functions by using your own user interface components. The user of the resulting application does not even need to know Datacap Mobile function is being used.

The SDK is available for iOS and Android.

iOS

For iOS development, the following is required:

- An Apple Mac
- Mac OS X Yosemite (10.10.X)
- Xcode 6.3+
- A working knowledge of Objective-C
- A Datacap 9.0 server

The SDK is written in Objective-C and is organized into three main modules:

- UI
- Model (Domain Layer)
- Network

There are two separate iOS frameworks:

Core	Network + model
UI	Controllers that use Core for some functions of the Capture app and that are generic enough to be displayed to third parties

The SDK is documented using AppleDoc. The SDK is located under Datacap’s installation directory, for example:

```
C:\Datacap\MobileSDK\iOSMobileSDK.zip
```

A sample app and HTML documentation are also provided (after the .zip archive is uncompressed):

```
iOSMobileSDK\iOS\Sample
\iOSMobileSDK\iOS\Documentation\html
```

Android

For Android development, the following are required:

- JDK 6 or later (JDK 7 for Android –5 Lollipop— and later)
- Android Studio, including Android Studio IDE and Android SDK (v 22 or later)
- Import the Datacap SDK project; will prompt to download additional SDK packages
- A working knowledge of Java
- A Datacap 9.0 server

The SDK is written in Java and is organized into packages for these tasks:

- Document object (domain) model
- Datacap service
- Image processing
- Recognition

The SDK is documented in Javadoc. The SDK is located under Datacap's installation directory, for example:

```
C:\Datacap\MobileSDK\AndroidMobileSDK.zip
```

A sample app and HTML documentation are also provided (after the Zip archive is uncompressed):

```
AndroidMobileSDK\Android\SampleApplication
AndroidMobileSDK\Android\Documentation
```

SDK objects

The SDK provides access to objects in the view (user interface), model (application objects), and network layers of Datacap Mobile. The objects that can be accessed and manipulated are shown in Figure 11-20. (Objects marked with an asterisk are expected to become available with a future release of the SDK, or to be provided by third-party developers and software providers.)

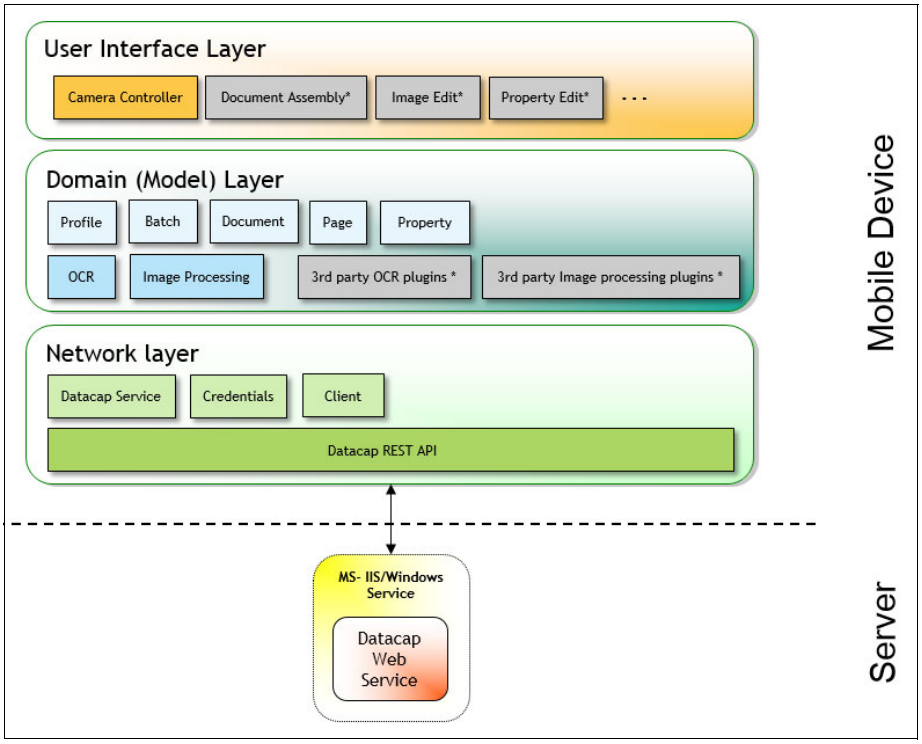


Figure 11-20 Datacap Mobile SDK objects

SDK object descriptions

This section contains additional information about each of the following SDK objects that are available to app developers:

- Datacap Service: Object that represents the Datacap server and connectivity to it: application ID, station ID, workflow ID, Job ID, Setup DCO name.
- Credential: Object to encapsulate the information required by the authentication method.
- Client: Object to interface with the Datacap service, which uses Credential. It gets the Setup DCO, monitors (retrieves) batch information, and uploads batches.
- Profile: Object to represent the Datacap Setup DCO (Datacap document hierarchy: a particular combination of batch, documents, pages and fields that have been defined for a given Datacap application). It returns the batch type, document types, pages types, and field types.
- Batch: Object instance of a Batch type to hold the contents of a batch. The Batch object is passed to the Client for upload to the Datacap Services.
- Document: Object instance of a Document type to hold the document data.
- Page: Object instance of a Page type to hold the page data.
- Field: Object instance of a Field type.

- OCR: OCR libraries included in the framework to recognize images associated with the Page objects.
- Image processing: Image-processing libraries that are included in the framework to perform all basic types of image transformation: edge detection, deskewing, rotation, filtering, and so on.
- Third-party OCR plug-ins: Plug-in architecture to support add-on OCR or ICR libraries to support future use cases, such as check recognition.
- Third-party image-processing plug-ins: Plug-in architecture to support future add-on image-processing libraries to support extended use cases, such as identification document recognition, check processing.
- Camera Controller: UI widget for manipulating the camera and automatic/manual capture
- Document Assembly Controller: UI widget to manipulate the structure of a batch (classify and assemble documents and pages).
- Field Edit Controller: UI widget to provide data entry capabilities associated with the Datacap document structure.
- Image Edit Controller: UI widget to provide image-processing capabilities, such as image cropping, rotation, conversion to black and white, and so on.

11.7 Representational state transfer (REST)

The Datacap Mobile application uses IBM Datacap Web Services. Interfacing to these endpoints over a network, and using them in a variety of combinations, enables an app to be constructed to work remotely with IBM Datacap.

For an introduction to representational state transfer (REST), read the IBM developerWorks article titled “RESTful Web services: The basics”:

http://www.ibm.com/developerworks/webservices/library/ws-restful/

If you are interested in exploring this topic, IBM provides a test tool to introduce Datacap Web Services:

http://ibm.co/1LOvg8W

Note: These endpoints are also available to third-party developers to create and build their own app or applications to interface with Datacap.

12

Customizing Datacap

You can use the FastDoc function directly to build you applications to a set of defined customer requirements. However, you might have requirements that are customer-specific and not available as standard. This could be a specific user interface (UI) layout or behavior. It might also be a specific third-party system that you want to integrate, but Datacap does not include a ready-for-use connector.

Datacap offers several flexible application programming interfaces (API) that enable partners and customers to build new functions or enhance existing ones of the core product. This chapter describes those and includes the following sections:

- Customizing Datacap Desktop
- Customizing the Datacap Desktop Scan panel
- Customizing the Datacap Desktop Verify panel
- Deeper Datacap Desktop integration

12.1 Customizing Datacap Desktop

IBM Datacap provides the Datacap Desktop client application for scanning and verifying document data and metadata. You can use Microsoft Visual Studio C# along with the IBM Datacap Desktop Developer Kit (DDK) to create custom verification and Scan panels for Datacap Desktop applications. This allows deep customizing of the look, feel, and function of your user interface.

The custom panel projects are available for download in two separate packages:

- Datacap Desktop Custom Panel Solution
- Datacap Desktop Universal Field-At-A-Time Panel

  The compressed file is called `DCDesktop-9.0-Panels-with-Universal.zip`, and you can find it by selecting **Download** → **Description** → **DCDesktop custom panel solution with universal**.

Both packages contain source code and instructions for the Visual Studio C# solution.

Datacap Desktop Custom Panel Solution contains the standard files for use in both Scan and Verify panels. Datacap Desktop Universal Field-At-A-Time Panel contains all of the same files, but also a modifiable version of the universal field-at-a-time Verify panel that can be used if no specific panel is defined in your application for use at Verify time.

You can find these packages in the Downloads section of the IBM Datacap 9.0 DDK Datacap Desktop Custom Panels web page:

http://ibm.co/1IzUqKB

As a prerequisite, IBM Datacap must be installed on your development system, along with Microsoft Visual Studio, to create or modify the Datacap Desktop panels. Based on information avaialble at the time of publication, we suggest using Microsoft Visual Studio 2013 for development of the panels. The `readme` files of the downloads provide additional information about hot fixes or other requirements.

These custom panel packages offer you several capabilities:

- Control for the layout of fields and images for Scan or Verify UIs
- Access to the Datacap Object (DCO) and the application variables
- Ability to build custom functions specific to your application needs
- Display of custom logos or graphics on your panel

12.2 Customizing the Datacap Desktop Scan panel

The purpose of a Scan panel is to manage the batch and provide a user interface (UI) to configure the scanner setting for capture of new images or adding new document files to a batch. The panel also provides controls for the operator to fix the batch by reorganizing pages and documents, assigning page and document types, and changing the batch in other ways that are required by the application.

Batch-level metadata can be entered from a Start Batch dialog panel and the display is based on specific task settings. The Start Batch panel is dynamically created, with data entry fields automatically displayed for all fields that are defined at the batch level within the application setup DCO.

Datacap Desktop includes several panels related to scanning (DotScanPanels), verification (DotEditPanels), and the Medical Claims application (MedicalClaimsPanels).

For scanning, the DDK provides four panels in the Visual Studio project under DotScanPanels:

ISISScan	Used to interface with the Image and Scanner Interface Specification (ISIS) driver of an attached physical scanner
TWAINScan	Used to interface with the TWAIN driver of an attached physical scanner
Vscan	Used to ingest files from a file system directory
startPan	Used to capture any required batch values/variables

Follow these steps to gain a high-level understanding of the capabilities available in the Datacap Desktop Scan panels:

1. Download both the Datacap Desktop Custom Panel Solution and the Datacap Desktop Custom Panel Solution with Universal packages, and extract them to suitable locations on your development system.
2. Download the relevant Microsoft Visual Studio application to provide a C# development environment.
3. Open Datacap Desktop Custom Panel Solution with Universal by double-clicking the **`DCDesktopPanels.sln`** file. This opens Visual Studio.

   All of the panels are listed in the Solution Explorer window, usually displayed on the right side of the Visual Studio IDE.
4. Double-click the **`ISISScan.cs`** file in the Solution Explorer to load the Layout view of the panel, which shows the layout of all the buttons, sliders, and fields.

   Change change the layout of the fields, the fonts used, and remove certain controls if you prefer.

   With C#, you can add functions to the panel as needed. For example, you can initially disable the Submit button, and then count the number of images actually scanned and compare it to the expected value entered. If they match, the Submit button is enabled and the process can be completed.
5. Double-click the **`TWAIN.cs`**, **`VScan.cs`**, and **`StartPan.cs`** files to view those layouts also.

12.2.1 Basic Scan panel customization example

As a simple example, add a fictitious company logo to the ISIS Scan panel to provide a custom look and feel to the panel.

> **Note:** Ensure that you have read any `readme` files associated with the DDK to verify that any relevant hot fixes have been added.

1. Create a folder named `Images` within your DDK application, and place your company logo image in this directory. For the purposes of this example, use a PNG file.
2. In Visual Studio, right-click the **`ISISScan.cs`** file and select **Copy**. Now press **Ctrl + v** to paste the copy into the solution. A new file called `Copy of ISISScan.cs` is created.
3. Rename the newly created file to `ISISScanLogo.cs`. You use this copy of the original in your custom application. Existing applications using `ISISScan` are not affected by any changes in this step.

4. Double-click the **ISISScanLogo.cs** file (Figure 12-1) to show the Layout view.

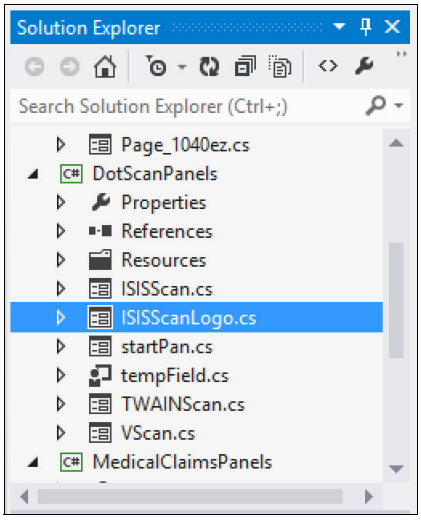


Figure 12-1 ISISScanLogo.cs file

5. Right-click the **ISISScanLogo.cs** file and select **View Code**. In the code, change any references to `ISISScan` to `ISISScanLogo`. If you do not do this, you get a compiler complaint later stating two files with the same name exist.
6. Scroll to the bottom of the window and expand the size of the panel by using your mouse.
7. Select all of the controls in this view by using Ctrl + A, and move them all down by using the mouse to make space for the logo.
8. Expand the control **Toolbox** on the left side and select the **PictureBox** control. Drag this onto the Layout window. Adjust the controls location and size accordingly.
9. Right-click the newly added **PictureBox** control and select **Properties** in the pop-up menu. The properties are shown on the right side of the Visual Studio IDE. Find the **Image** property, and click the link icon to select an image to display.
10. In the Select Resource window, click **Import**. Now, select the file that you copied to the image folder earlier and click **OK**. See Figure 12-2.

    Repeat this process for the InitialImage property.

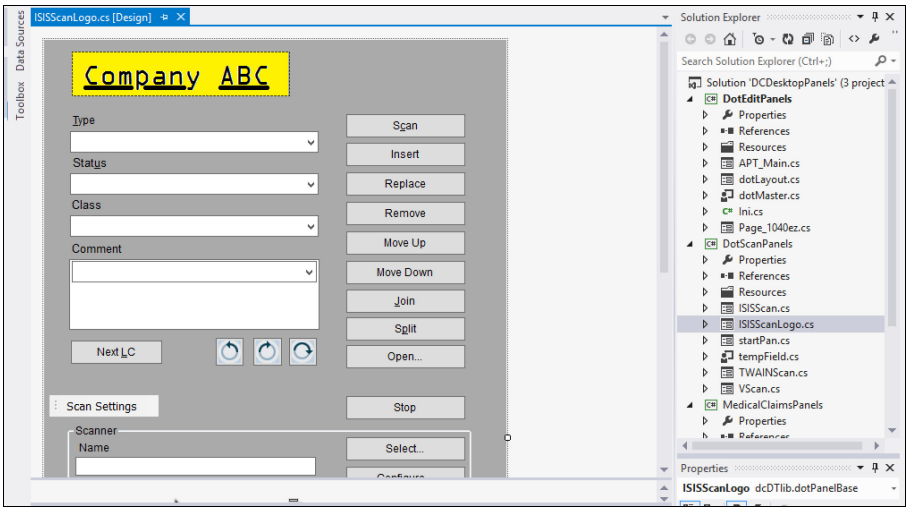


Figure 12-2 Import file

11. The image appears on the scanner panel layout. Make any needed size or location adjustments and save the panel.
12. Compile the panels for use in our Datacap Desktop applications. In the Solution Explorer window, under the DotScanPanels project, double-click **Properties** to load various settings. Select the **Build** tab to load the build settings and values.
13. Go to **Output Path** and click **Browse**. Note the location that this is pointing to.
14. Compile the application by pressing the F7 key or selecting **Build** from the Build menu.
15. Make a copy of the `C:\Datacap\dcDesktop` folder as a backup of the original settings and configuration for Datacap Desktop.
16. Go to the **Output** directory that was noted earlier,and copy the **`DotScanPanels.dll`** file to the `C:\Datacap\dcDesktop` folder, replacing the original one.
17. Next, configure your application to use this new panel.

    Log in to the Datacap web client to modify your workflow and configure a task to use the new panel.
18. Go to **Administration** → **Workflow** and modify an existing scan task in your workflow, that is to use the new custom panel.
19. Ensure that the Program is set to either **DCDesktop** or **Multiple**.
20. Click **Setup**. In the DCDesktop section, change the User Interface Panel to the name of the panel (ISISScanLogo) that was just created, and click **Save**.
21. In the Datacap web client, create a shortcut for testing. Select the **Shortcuts** tab, create a new shortcut, and assign it to the batch creation task. Make it **Auto** and click **Apply**.
22. Ensure that the user has relevant permission to use this shortcut and task.
23. Test your new panel by starting **Datacap Desktop**, and selecting the shortcut that contains the new task. Then, select **Run Pending**.

    The new UI should be rendered for use. See Figure 12-3.

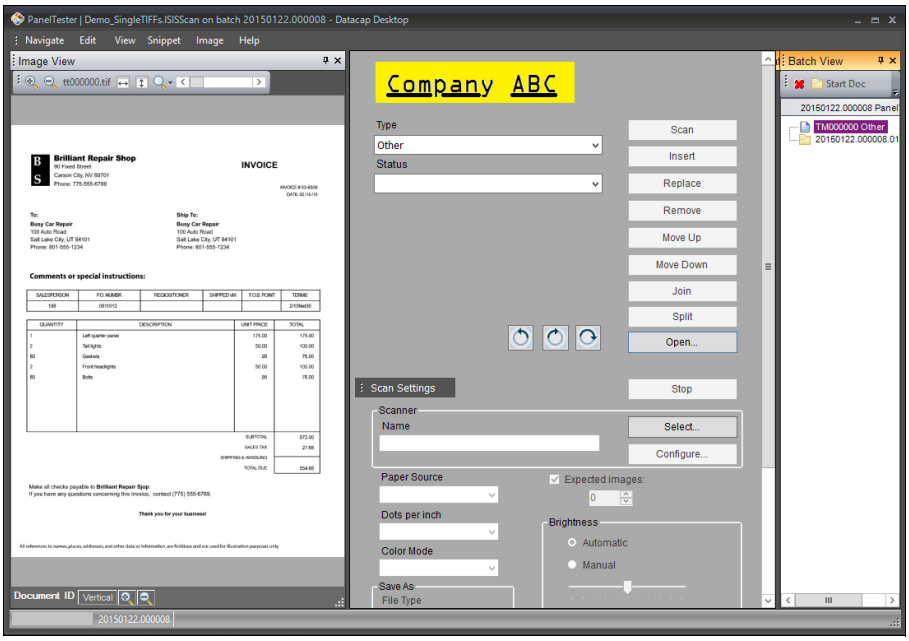


Figure 12-3 UI with new logo

12.3 Customizing the Datacap Desktop Verify panel

The purpose of a Verify panel is to operate on the batch being processed and provide a UI to validate and verify the data captured. The panel can also provide controls for the operator to fix the batch by reorganizing pages and documents, assigning page and document types, and changing the batch in other ways that are required by the application.

Datacap Desktop Developer Kit includes several panels that are related to verification:

- The *APT* custom panel is the base panel that is used by the Accounts Payable application.
- *MedicalClaimsPanels* contains the panels that are used by the Medical Claims application.
- *1040EZ* is the panel that is used with the sample 1040EZ application that is available on developerWorks.
- The *UniversalPanel*, or *field-at-a-time* panel, works with any application as the default panel.

All of the Datacap Desktop Verify panels are in the DotEditPanels Project except the Medical Claims panels, which are in the MedicalClaimsPanels project.

12.3.1 Basic Verify panel customization example

As a simple example, we add a fictitious company logo to the Verify panel for a sample application. We also add an advanced snippet capability to allow use of one large snippet for specific fields to give a custom look and feel to the panel. Although It is possible to add a validation process to the panel, the validation routines should be built by using rules in either FastDoc or Datacap Studio when possible.

> **Note:** Ensure that you have read any readme files associated with the DDK to ensure any relevant hot fixes have been added.

Follow these steps to customize the Verify panel:

1. Create a folder called `images` within your DDK application. Place your company logo in this directory. For the purposes of this example, we use a PNG file.
2. Ensure that the Datacap Desktop is not open.
3. Open Visual Studio and load the Datacap Desktop DDK. Ensure that DotEdit is set as the startup project by right-clicking **DotEdit** in the Solution Explorer window and selecting **Set as Startup Project**.
4. Press Ctrl+F5 (Start without Debugging) to run the dotMaster panel (Figure 12-4 on page 283). In that panel, you can build layouts based upon the DCO or XML file of the existing Datacap applications.

   DCO Setup is used to load the Datacap Object of an application. The Datacap Object holds the definition of all Documents, Pages, and Fields for a specific application that you created in FastDoc or DStudio.

   Layout XML is used for migrating applications created in Datacap V8.0 and earlier using Datacap Batch Pilot. Therefore, we leave this blank for this project.

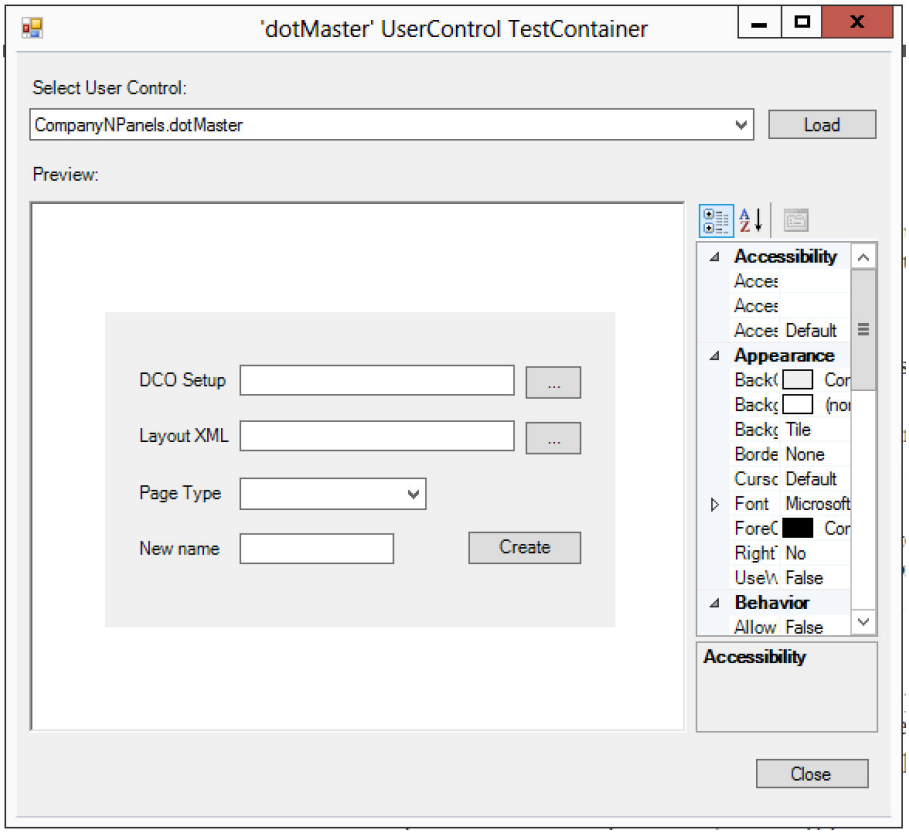


Figure 12-4 dotMaster panel

5. On the dotMaster panel, click **Browse** next to the DCO Setup field. Select the file setup DCO for your application, and click **OK**. For example, the TravelDocs DCO is typically in `C:\Datacap\TravelDocs\dco_TravelDocs\TravelDocs.xml`.
6. For this example, we have a test application called **PanelTester**, which has a single page document that has three associated fields.
   a. In the `New name` field, enter the name for the new C# class or the page name that you want to call it within your Visual Studio project. By default, it is the same name as the page type. However, as a preferable practice and to differentiate, prefix it with `Custom_`:

      `Custom_Flight.cs`

      > **Note:** If you create a new panel for the same DCO page, be sure to give it a different name. Otherwise, the original one will be overwritten within the Visual Studio project.

   b. Click **Create**.
7. Visual Studio displays a message to indicate that you must reload the project. Click **OK** to close the message window, and then click **Close** to shut the UserControl window.
8. Next, Visual Studio asks whether you want to reload. Agree by clicking **Reload All**.

9. When you see Custom_PanelTester (or another chosen name) appear in the Solution Explorer window, double-click **`Custom_PanelTester.cs`** to display the new custom panel design for the application. See Figure 12-5.

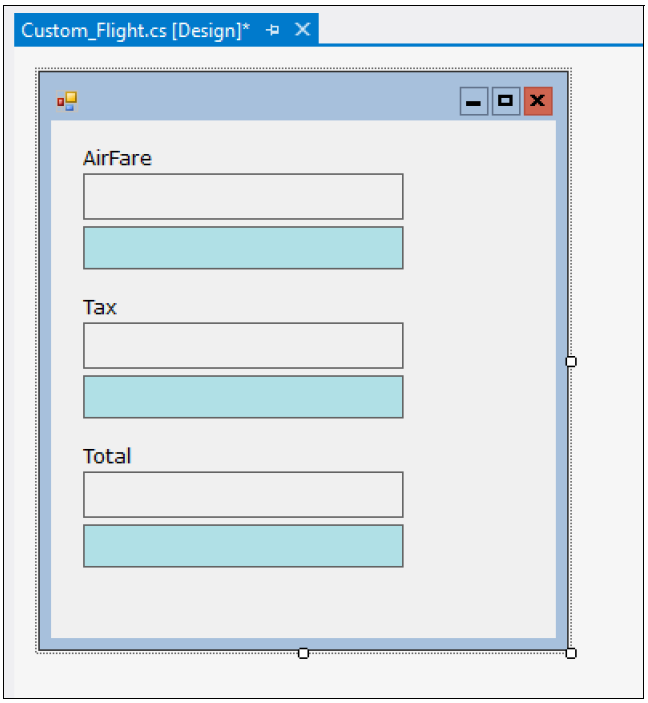


Figure 12-5 New custom panel design

10. Use your mouse to drag and resize the fields as necessary.
11. Scroll to the bottom of the window and expand the size of the panel using your mouse.
12. Select all the controls in this view by using Ctrl + A, and move them all down by using the mouse to make space for the logo.
13. Expand the control **Toolbox** on the left side and select the **PictureBox** control. Drag this onto the Layout panel. Adjust the controls location accordingly.
14. Right-click the newly added **PictureBox** control and select **Properties** in the pop-up menu. The properties are now shown on the right side of the Visual Studio IDE. Find the Image property, and click the link icon to select the image that you copied earlier.
15. In the Select Resource window, select **Local Resource** and click **Import**. Select the file that you copied to the `image` folder earlier, and click **OK**. Repeat this process for the InitialImage property.
16. The image now appears on the scanner panel layout. Make any needed size or location adjustments and save the panel.
17. Click one of the fields labels, and change the size of the font in the Properties window from Verdana 9pt to **Verdana 12pt**.
18. Select the first snippet object on your panel, copy it (Ctrl + C), and paste it (Ctrl + V). Move or resize the snippet as needed.

    Notice the name of the snippet in the Properties tab, which is typically `axDcimage1`. You will need this later.

19. Click the first field in the list that you want to work with, and select the **Enter** event in the Properties panel on the right side of the window (you might need to click the lightning bolt icon to reveal this option). Double-click the **Enter** field to open a code window.

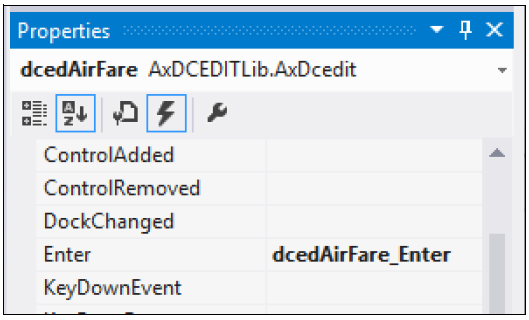


Figure 12-6 Properties

20. You then see an empty method similar to Figure 12-7.

```
private void axDcimage1_Enter(object sender, EventArgs e)
{

}
```

Figure 12-7 Empty method

Paste the following code into the method:

```
AxDCEDITLib.AxDcedit pEdit = this.ActiveControl as AxDCEDITLib.AxDcedit;

if (pEdit != null)
{
XmlNode BoundField = pEdit.Tag as XmlNode;

if (BoundField != null)
{
PopulateLineSnippet(BoundField, axDcimage1);
}
}
```

Note: The second argument in PopulateLineSnippet must match the name of your new superSnippet object (Step 18 on page 284), which is axDcimage1.

21. The PopulateLineSnippet method is underlined in red to alert you about not existing in the current context. Do not worry about this for now.

22. Follow the same process for adding the code to the Enter event for all remaining snippets on the page (see Figure 12-8).

```
private void axDcimage1_Enter(object sender, EventArgs e)
{
    AxDCEDITLib.AxDcedit pEdit = this.ActiveControl as AxDCEDITLib.AxDcedit;

    if (pEdit != null)
    {
        XmlNode BoundField = pEdit.Tag as XmlNode;
        if (BoundField != null)
        {
            PopulateLineSnippet(BoundField, axDcimage1);
        }
    }

}
```

Figure 12-8 Adding code to the Enter event

23. Next, add the code shown in Example 12-1. This code runs when the snippet of the current field that you are working on is entered. It grabs the coordinates of the snippet and displays them in the new larger or central snippet. After it is added, you see that the issue of PopulateLineSnippet is resolved.

Example 12-1 PopulateLineSnippet sample code

```
private void PopulateLineSnippet(XmlNode BoundField, AxDCIMAGELib.AxDcimage
ImageCtrl)
    {
        string sPos;
        Int32 nL;
        Int32 nT;
        Int32 nR;
        Int32 nB;

        XmlNode LineField;
        LineField = BoundField;
        //Get the LineField coordinates
        sPos = LineField.SelectSingleNode("V[@n='Position']").InnerText.ToString();
        string[] arPos = sPos.Split(',');
        string dispString = arPos[0] + "," + arPos[1] + "," + (Convert.ToInt32(arPos[2]) -
Convert.ToInt32(arPos[0])).ToString() + "," + (Convert.ToInt32(arPos[3]) -
Convert.ToInt32(arPos[1])).ToString();
        ImageCtrl.DispZoneString = dispString;

        sPos = BoundField.SelectSingleNode("V[@n='Position']").InnerText.ToString();
        arPos = sPos.Split(',');
        Int32.TryParse(arPos[0], out nL);
        Int32.TryParse(arPos[1], out nT);
        Int32.TryParse(arPos[2], out nR);
        Int32.TryParse(arPos[3], out nB);
        ImageCtrl.EraseRect(-1);
```

```
        ImageCtrl.DrawRect(nL, nT, nR, nB, 2, 200 << 16 | 200 << 8 | 255, 200 << 16 | 200 << 8 | 255,
"");
    }
```

Figure 12-9 shows the completed layout of the customized Verify panel.

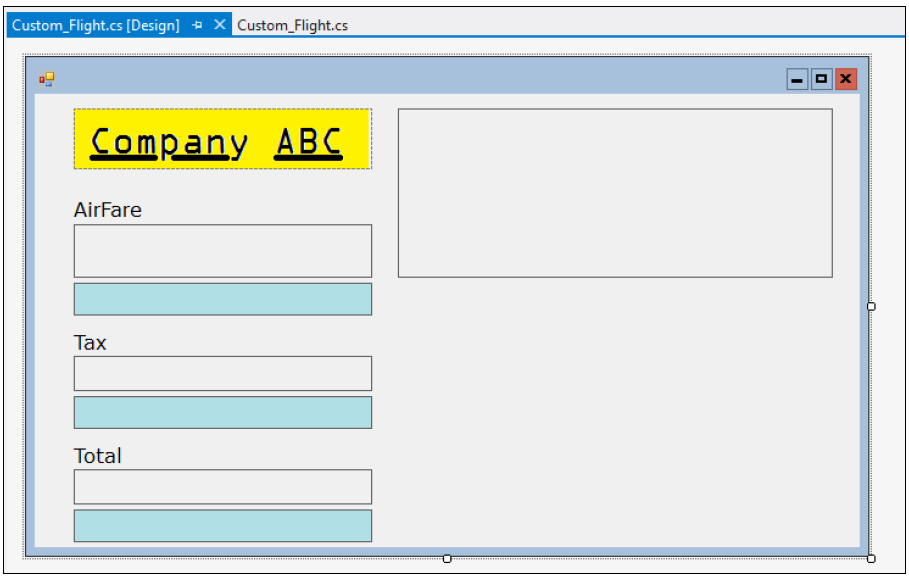


Figure 12-9 Customized Verify panel

24. Compile the panels for use in our Datacap Desktop applications. In Solution Explorer window, under the DotScanPanels project, double-click **Properties** to load various settings. Select the **Build** tab to load the build settings and values.
25. Change the Output path to point to `C:\Datacap\DCDesktop` and save the settings.
26. Build the application.
27. Start the Datacap web client and log in to the Administrator UI for the application you are working on.
28. Select **Workflows**, and then select the workflow name to work with, **PanelTester**, and click **Edit**.
29. Select the job to work with, **Verify_Export**, and click **Edit**.
30. Click the **Tasks** tab, select **Verify**, and then click **Edit**.
31. Confirm that the program being used is DCDesktop or Multiple.
32. Under the Advanced tab, search for the **Datacap Desktop** section.
33. Enter the DCO page type, `main_page`**,** that you want to verify in the panel for fields.
34. Enter the C# class, `DotEdit.Custom_Flight`, that you generated for the page type in your Visual Studio project (Figure 12-10).

Figure 12-10 Build DCO type to panel

35. Click **Save** and **Close**.

36. Next, open **DCDesktop**, log in to your application, and run it to the Verify stage. You should see the new panel rendered in the UI.

37. As Figure 12-11 shows, the new, larger snippet that was created earlier gets populated with the snippet of the relevant field we are working with, but in a larger area. This makes it easier for the user to work with.

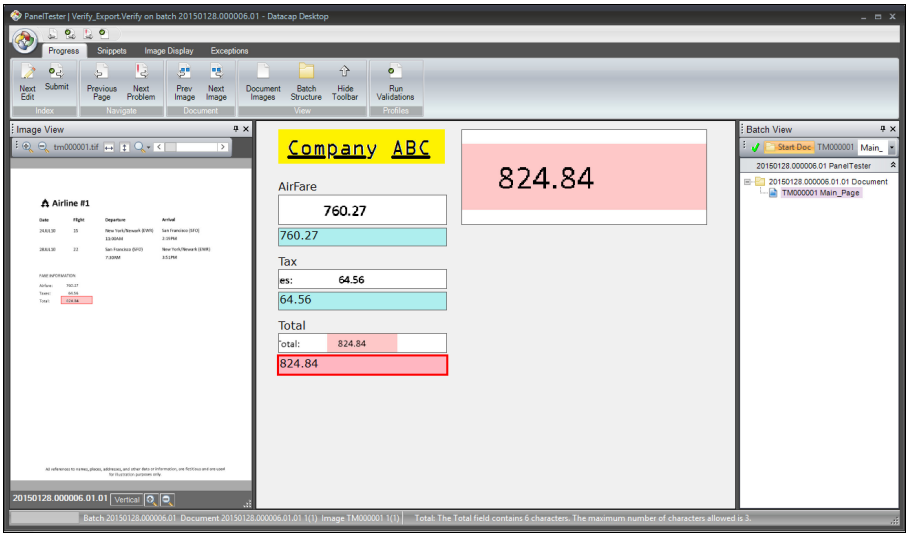


Figure 12-11 Modified panel

12.4 Deeper Datacap Desktop integration

There are a wide range of supported API calls available for the different panels. These range from adding pages, working with thumbnails, checking the document structure, validation of data, scanner settings, and so on. For more information, see the documentation that is included as part of the Datacap Desktop Developer Kit.

12.4.1 Datacap action customization using the Datacap Object API

When you cannot deliver application functions by using read-for-use library actions, you can write custom actions to fulfill your processing needs. IBM Datacap enables development of actions in both VBScript and Microsoft .NET C# through the Datacap Object API.

These are a few action development examples:

- Integrating additional OCR engines
- Interfacing to third-party systems
- Complex validation routines
- Complex DCO restructuring and parsing

Actions are small snippets of code that are wrapped into reusable containers that you can easily add to your Datacap application by using Datacap Studio. This capability hides some of the more complex coding routines from the capture developers, leaving them free to concentrate on the process rather than decoding lines of code.

To create custom actions, you need to be experienced with Datacap Studio, Datacap document hierarchies (DCO), C# programming, and XML.

Note: Any custom actions created for previous versions of Datacap must be updated to run with IBM Datacap 9.0. This version does not require DLL registration and does not require a separate `RRX` file to be installed with the DLL.

For more information about Datacap custom actions, see these web pages:

- IBM Datacap 9.0 DDK custom actions

  http://ibm.co/1IzUutT

- Datacap object API reference

  http://ibm.co/1Ph5w9J

C# example

In this example, we build a simple action in C# to grab a value from a field and check wheter the `regex` matches value. If it matches, the action returns `true`. If it does not match, the action returns `false`.

First, get the Custom Actions template from the Downloads section of the Custom Actions web page:

http://ibm.co/1IzUutT

Note: The download includes instructions for how to migrate actions from Datacap 8.1 to Datacap 9.0. That information is beyond the scope of this book.

Follow these steps to build an action in C#:

1. Copy the **`Datacap 9.0 NET 4.0 Action Template.zip`** file that you downloaded into the template directory for Microsoft Visual Studio:

   ```
   C:\Users\p8admin\Documents\Visual Studio 20012\Templates\
   ProjectTemplates\ Visual C#
   ```

2. Start Visual Studio and select **New Project**.
3. Click **Installed** → **Templates** section, and then select **Visual C#**.
4. Select **Datacap 9.0 NET 4.0 Action Template**.
5. Enter `customValidation` for the project name, and click **OK** to create the project.
6. In the Solution Explorer window, double-click the class that was just created (**customValidation.cs**) to display the code.
7. Expand **CustomActions** in the code window to view the sample action that is provided as part of the project.
8. Build the project by either pressing F7 or selecting **Build Solution** from the Build menu.
9. Expand **Properties** in the explorer, and check for any references with yellow exclamation marks. Delete these references by right-clicking and selecting **Remove**.
10. Re-add the reference by right-clicking **References** and selecting **Add Reference** from the pop-up menu.
11. For the DCSmart reference, choose the **Browse** tab, and then browse for the removed `C:\Datacap\DCShared\Net\` and select **DCSmart.dll**. Click **OK**.
12. For the iRRX reference, choose the **Browse** tab and then browse for the removed `C:\Datacap\DCShared\Net\` and select **iRRX.dll**. Click **OK**.
13. Use the COM tab to find the references for dclogXLib, dcrroLib, PilotctrlLib, and TDCOLib.

14. Rebuilding the project resolves the reference, and the DLL is built.

    We suggest testing with the sample action to ensure all is working before adding any new code to the project.

15. Copy the DLL that was created, that is, `C:\Users\p8admin\Documents\Visual Studio 2012\Projects\customValidation\customValidation\bin\Debug` in to the Rules directory of your application, that is, `C:\Datacap\PanelTester\dco_PanelTester\rules`.

16. Open DStudio, load the application that you intend to use this with, in this case, PanelTester, and you see the library load the actions into the Application Specific Action area of the Action Library window (Figure 12-12).

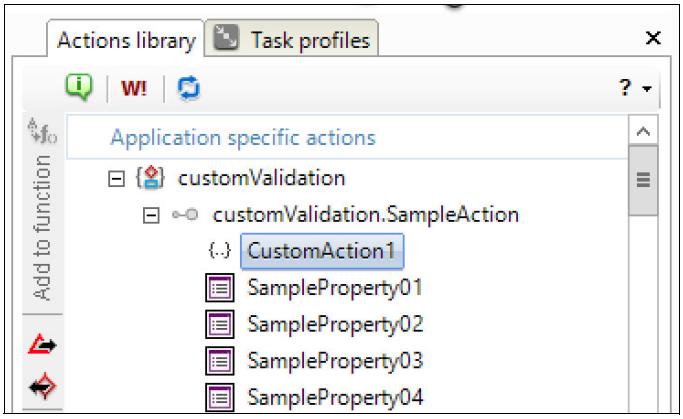


Figure 12-12 Actions library

17. The action is now ready to use in your Datacap application.

18. Next is to build your own action.

19. Copy the code in Example 12-2 into the customValidation.rrx file in Visual Studio. Ensure that you include **`System.Text.RegularExpressions`** at the top of the file.

Example 12-2 Sample C# action code

```
using System.Text.RegularExpressions;


……

public bool CustomIsFieldMatchingRegex(string p1)
        {
            bool bRes = false;      // intially set the retun result to false and only
set to true if the action is successful

            try
            {
                string fieldValue = CurrentDCO.Text.ToString();
                WriteLog("The current field : " + CurrentDCO.Type + " has a value of " +
fieldValue);
                WriteLog("Checking the field with regex : " + p1);

                Regex regex = new Regex(p1);
                Match match = regex.Match(fieldValue);
                if (match.Success)        // if the regex is found then return true
                {
                    WriteLog("Match for regex found");
                    bRes = true;
```

```
                }
                else
// regex was not found
                {
                    WriteLog("Match for regex not found");
                    bRes = false;
                }

            }
            catch (Exception ex)
            {
                // It is a best practice to have a try catch in every action to prevent
any unexpected errors
                // from being thrown back to RRS.
                WriteLog("There was an exception: " + ex.Message);
            }

            // return the reuslt of the action back
            return bRes;
        }
```

20. Next, update TheRRX.rrx file under Resources in the Solution Explorer window. Add the XML in Example 15-3 to the `RRX.rrx` file. This is used to define the information for the action and also display it in the Datacap Studio action library.

Example 12-3 Action XML example

```
<method name="CustomIsFieldMatchingRegex">
    <p name="p1" type="string" qi="Enter the regex you wish to test against"/>
    <ap>
      The regex to be checked against the Field object's value.
    </ap>
    <h>
      Determines if the regex entered as the parameter matches the captured value of the
current Field object.<br/><br/>
      <e>
        <b>CustomIsFieldMatchingRegex([0-9]{3})</b><br/>

      </e>
    </h>
    <lvl>Field levels</lvl>
    <ret>
      <b>True,</b> if the action succeeds.  Otherwise, <b>False.</b>
    </ret>
  </method>
```

21. **Save** and **Build** the project.
22. Copy the newly built DLL in `C:\Users\p8admin\Documents\Visual Studio 2012\Projects\customValidation\customValidation\bin\Debug` to the application rules folder, `C:\Datacap\PanelTester\dco_PanelTester\rules`. You might need to exit Datacap Studio before copying because Datacap places a lock on the DLL.
23. Reopen DStudio, load the application that you intend to use this new custom action with, in this case Panel Tester, and you see the library loads the new action into the Application Specific Action area in the Application Library window.

24. You can now use this action in your applications. See Figure 12-13.

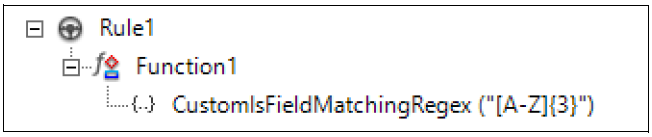


Figure 12-13 New action

This is a basic action. However, the Datacap Object API allows granular use of the DCO to perform detailed customization and to permit use of third-party API for integration purposes. For more information about the API, see “Datacap object API reference” in the IBM Knowledge Center:

http://ibm.co/1Ph5w9J

12.4.2 Customizing Datacap ruleset templates

IBM Datacap applications are built using rulesets and actions that contain the process definitions performed by an application. You build a ruleset by adding one or more actions to a function within a rule. This delivers the logic and functions for your capture process. In the past, Capture developers needed to understand how to arrange to use and arrange these actions and their various arguments.

In Datacap 9, you can now use the Datacap C# ruleset template in Microsoft Visual Studio to create a Ruleset Configuration Panel. This is a UI which significantly simplifies development. This is achieved by grouping appropriate actions into rules and exposing them through a simplified User interface. This makes changing setting much simpler and aids the Capture developer in understanding the process. The ultimate aim is to speed up and simplify development time.

The panel can be displayed in either FastDoc or Datacap Studio and can be customized to provide configuration selections that dynamically creates a ruleset to be run within an application.

Several ready-for-use Ruleset Configuration Panels are available from the product, however, there might be a call for development of a custom panel to meet specific requirement for a customer.

Note that a Ruleset Panel provides an easy way to configure rules. However, creation of a Ruleset Configuration Panel is not straight forward and can take longer to build and implement than manually creating a standard ruleset in Datacap Studio. Therefore, consideration should be made as to whether creation of a Ruleset Panel is needed in certain situations.

We now take the example ruleset and configure it for use in a Datacap application.

First we download the custom actions template from the IBM Datacap 9.0 DDK: Customizing ruleset configuration panels for FastDoc and Datacap Studio web page:

http://ibm.co/1TlH9ID

Follow these steps to configure a ruleset for a Datacap application:

1. Copy the **`Datacap 9.0 RRS Ruleset Template WPF.zip`** file that you downloaded into the template directory for Microsoft Visual Studio:

   `C:\Users\p8admin\Documents\Visual Studio 20012\Templates\ProjectTemplates\Visual C#`

2. Start Visual Studio and select **New Project**.
3. Select **Installed** → **Templates** section, and select **Visual C#**.
4. Select **Datacap RRS 9 Ruleset1**.
5. Enter **customEmailPanel** for both the project and solution name and click **OK** to create the project.
6. Expand **References** in the solution explorer and check for any references with a yellow exclamation mark. Delete these highlighted references by right-clicking and selecting **Remove**.
7. Readd the reference by right-clicking **References** and selecting **Add Reference** from the pop-up menu.
8. Navigate to find the reference and add it. There are two COM references, `DCApple 1.0` and `TDCO 1.0`. The third reference is `iRRX.dll`, which is typically found in `C:\Datacap\dcshared\NET`.
9. The Embed Interop Types property for DCAppleLib and TDCOLib must be set to **True**. The Embed Interop Types property for iRRX must be set to **False**.
10. Build your project. Create a DLL file called `customEmailPanel.rul.dll`. This will be in a directory similar to `C:\Users\p8admin\Documents\Visual Studio 2012\Projects\customEmailPanel\customEmailPanel\bin\Debug`.
11. Place the **`customEmailPanel.rul.dll`** and **`customEmailPanel.Rul.dll.config`** files in the Datacap RRS directory. Typically, the location is `C:\Datacap\RRS`.
12. Open Datacap Studio, and load your sample project, **PanelTester**. You see your new panel loaded in the Global Ruleset panel.
13. Right-click the newly created **Sample Email** ruleset, and select **Install in Application**. This action copies the `.dll` and `.config` files to the local rules folder of your application. It also updates the `collection.xml` file to reflect this addition.

14. Click the newly added **Sample Email** ruleset, and then click **Setting** in the Properties tab to display the settings (Figure 12-14).

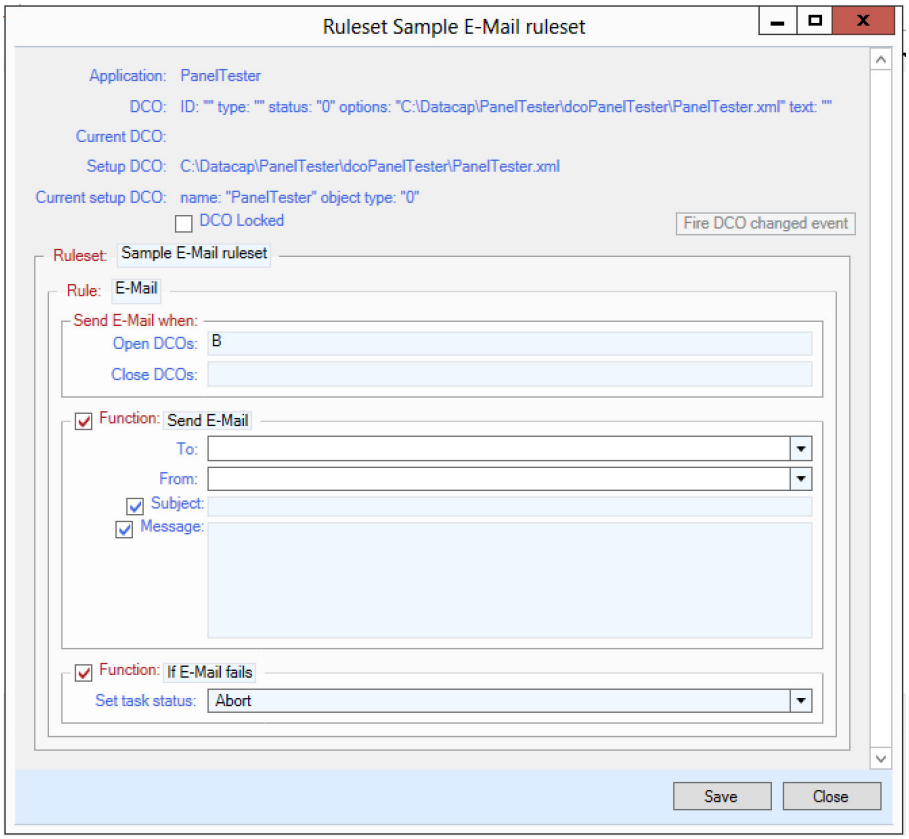


Figure 12-14 New ruleset

15. Enter various values as needed for To, From Subject, and Message, and then add it to your project and run the ruleset. This should open your email client and send an email message, as defined in the ruleset.

13

Datacap scripting

This chapter describes how to enhance IBM Datacap applications by using actions that you create yourself. We refer to this as *scripting* because such actions are created within the Datacap application. This enables you to quickly and easily extend Datacap capabilities as dictated by your business needs.

This chapter covers the following topics:

- Introduction
- The basics of actions
- Getting started
- Documenting your action
- Writing an action
- Referencing other objects from DCO or CurrentObj

13.1 Introduction

Although most processing of a capture application can be accomplished by using actions that are included with the product, needs often arise that require you to write an action that is tailored to a specific business requirement. These *custom actions* can be stored in libraries and reused whenever you need them. The difference between a good capture application and a great one is often the inclusion of a few well-placed, custom actions.

For example, let's say that you are capturing a document containing a 15- or 16-digit credit card number. Using standard validations, you can read the number and ensure that it is a minimum of 15 digits, a maximum of 16 digits, and 100% numeric. This validation can be made considerably better by applying an algorithm called *Luhn Mod 10*.

With a little effort, you avoid the following problems:

- The recognition engine misreads a value, yet does not flag it as low-confidence and alert the data entry operator to check it (this is called a *substitution error*).
- The recognition engine reads data correctly but flags one or two characters as low-confidence. This causes the field, and possibly be the entire document, to be viewed unnecessarily by the data entry operator, which creates more work. Worse, if the application flags too many fields that are actually correct, data entry operators might be lulled into assuming that the flagged values are correct (this is called a *false positive*). Flagging too much or too many characters is as bad as not flagging any.

There is currently no ready-for-use action to perform Luhn Mod 10 validation. To take the capture and validation of this field from a good level to a great level requires obtaining or writing a custom action to do that.

Scripting is not used only when writing a custom action, however. There is at least one ready-for-use action that enables you to process documents without writing a complete action (`ProcessChildren`), and the Flex application has a facility that enables you to script without writing your own action (the *inline code* column in the Flex database). These are powerful tools, and others can be added in the future.

13.2 The basics of actions

Actions are the most basic building blocks of a Datacap application. They perform specific tasks, such running optical character recognition (OCR), connecting to a database, or returning information about a field. In essence, an action is a function written in Visual Basic Script (VBScript) or C#, which are part of the Microsoft .NET languages. It is important to realize that an action in either of these languages can call code objects written in either languages.

13.2.1 How actions are used

Actions are often grouped with other actions in a structural code device called a *rule function*. It might be confusing that an action is a function and that a grouping of actions in a Datacap rule is also called a function. But in programming, functions can call smaller functions so that they work as a group or a larger building block. In housing construction, nails and beams can be put together to form rafters. Rafters and other materials are put together and called a roof. A roof and other materials can form a house. It is just a process of putting smaller things together to make something larger.

The credit card number mentioned earlier provides an example of combining ready-for-use actions to accomplish a larger objective. Example 13-1 is a function that performs basic validation of a credit card number.

Example 13-1 Credit card validation function 1

```
Rule: Validate CC Number
      Function 1:
          IsFieldLengthMin(15)
          IsFieldLengthMax(16)
          IsFieldPercentNumeric(100)
```

Using the three actions together performs basic validation of the field, but there might be other acceptable values for the field also. For instance, you might want to allow a credit card value to be blank because the document includes a check for payment rather than a credit card. You would not want to set the `IsFieldLengthMin` to `0`, because there are many numeric values between 0 and 16 characters that would never be acceptable as a credit card number. In such cases, you can add a function that can be satisfied for validation purposes, as shown in Example 13-2.

Example 13-2 Credit card validation function 2

```
Rule: Validate CC Number
      Function 1:
          IsFieldLengthMin(15)
          IsFieldLengthMax(16)
          IsFieldPercentNumeric(100)
      Function 2:
          IsThisFieldEmpty()
```

Each action in Example 13-2 returns TRUE or FALSE. If all of the actions in a function return TRUE, the rule is considered TRUE and it stops processing the rule. For instance, if you have a number such as 1234567890123456, the function checks to see whether it has a length of at least 15 characters (TRUE). Next, it checks whether the length is at or under the maximum of 16 (TRUE). Lastly, it checks the value to see whether it is 100% numeric (TRUE) and the rule is considered TRUE. Then the credit card number has been successfully validated.

If any of the actions in the first function return FALSE, it starts processing the second function. For example, this happens if the value being evaluated is in fact a blank value. It checks the minimum length of 15 and returns FALSE. It then processes actions in the second function. The action in the second function returns TRUE, which also makes the rule TRUE. The credit card number is successfully validated by the `IsThisFieldEmtpy()` action in Function 2.

If a value such as `012345` or `AB2345` is processed, it ultimately fails an action in both functions, and the rule is considered FALSE. In that case, the credit card number fails validation and is flagged for verification by an operator.

13.2.2 Type versus ID

Rules are applied to an object *type*. Each object (batch, document, page, and field) has two attributes: Type and ID. The type is the generic name that can apply to many separate objects in a batch. For instance, your batch might contain many pages of the type *Main_Page*. Attaching a rule to Main_Page causes it to run on every Main_Page in your batch.

The ID is a specific name that identifies one specific object within the parent object. In other words, every document in a batch has an ID that is different from every other document, even though they might all be of the same object type.

Bear in mind that rules are attached according to type, not ID.

13.2.3 True or False

It is important to understand that actions can return only TRUE or FALSE. Today, the preferred practice is for actions to always return TRUE unless they are used to make a *decision*; that is, unless they are used as a Boolean expression. Actions with names that contain “Is” (*IsFingerPrintClass*), “Check” (`CheckDocCount`), or “Compare” (`rrCompare`) reflect this behavior in their names. If the return value is FALSE, such decision actions cause a trailing function to run, as intended.

Other actions, which are used to *run* a command (for example, connect to a database, run OCR, set a variable) should return TRUE even if they encounter an error. Rather than returning FALSE, the error is written to the log file for debugging and testing purposes. Otherwise, handling exceptions makes the Datacap application unnecessarily verbose and complex.

For historical reasons, not all Datacap ready-for-use actions adhere to the preferred practice of always returning TRUE. For details about what each action returns, see the action’s documentation by right-clicking it in the Datacap Studio Action library and selecting **Information**. See also section 13.4, “Documenting your action” on page 302 for information about creating documentation for your own actions.

An example that demonstrates why carefully considering the return value is important is the Datacap global database connection, which allows actions to access a database by using the `OpenConnection()` action. This can fail if a previous `OpenConnection()` action was applied and the database it connected to was not closed before attempting to open another connection through the same connection object.

The obvious thing to do would be to always call the `CloseConnection()` action before attempting to open a new connection. However, trying to close a previous connection might also cause an error. For instance, the connection might not have been used previously, so any attempt to close it would not be able to run. If coded so that the `CloseConnection()` action returns FALSE if there is no open connection, opening a connection is done using the functions and actions as shown in Example 13-3.

Example 13-3 Open database connection with condition check

```
Function 1:
    CloseConnection()
    OpenConnection(<connection string>)
Function 2:
    OpenConnection(<connection string>)
```

Today, the `CloseConnection()` action just writes to the log that it could not close the previous connection because there was no open one, but it returns TRUE. That allows us to dispense with writing the second function and simplifies the code, as show in Example 13-4.

Example 13-4 Open database connection without condition check

```
Function 1:
    CloseConnection()
    OpenConnection(<connection string>)
```

Current practice calls for all actions to always return TRUE and to log any “errors” in the log file unless the action is used in a situation where you want it to move on to the next function.

13.2.4 Three styles

There are three styles currently in use with Datacap scripting. The majority of the read-for-use actions are in what we call the “old style” format. These functions are denoted by a gray diamond icon in the Library selection.

Old-style actions are written in VBScript and accept either no parameter or a single parameter, depending on the action. Even with a single parameter, you can specify multiple values. With few exceptions, the multiple values are separated by commas. See the `dcpdf_SetImageResolution` in Figure 13-1.

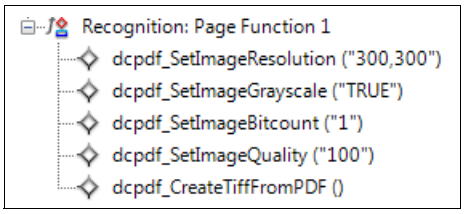


Figure 13-1 Old style actions

When IBM purchased Datacap and Datacap became a worldwide product, old-style actions that accept a floating point number as a parameter were rewritten in what we call the “new style” action format. This is because, in many countries, floating point numbers use a comma for the decimal separator, and that causes problems for old-style actions that require multiple comma-separated values. New-style actions are denoted by a red triangle icon (Figure 13-2). You often find two actions with similar names, one written in the old style (for compatibility with an earlier version) and one in the new style. New-style actions are not limited to a single parameter, and in some versions of Datacap, they can be *typed* so they accept only a certain data type. Typing is not common because of the need for actions to run under all versions of Datacap.

Figure 13-2 New style action

Old- and new-style actions are both written in VBScript, and you can see the VBScript code that they are composed of. The libraries are contained in the RRS folder. Some libraries have an `RRX` extension, and those can be opened in a text editor and can even be modified. However, if you modify them, it is a good practice to copy them into the rules folder of your application to prevent future installs of Datacap from overwriting your work. Files in the rules folder are used in preference to ones with the same name in the `RRS` folder.

Several libraries in the `RRS` folder with a `.dll` extension, such as `invoice.dll`, can also be opened in a text editor. However, there is some binary header information in these files, and they cannot be edited. You can certainly view the files to learn from them and even copy actions from them and place them in your own libraries for modification.

The third type of action is one that is written in Visual C#, and those actions have this icon next to them: {..} (see Figure 13-3). The source code for these actions is not provided to the public.

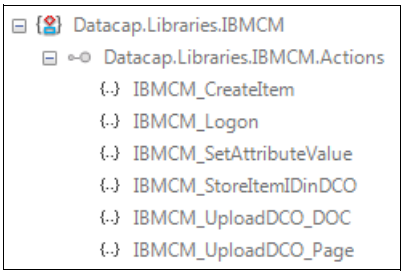


Figure 13-3 Visual C# actions

13.2.5 Actions can call other actions

Because new- and old-style actions are actually VBScript functions and because VBScript functions can call other functions, you can call VBScript actions from other VBScript actions. This is handy if you encounter an action that, for instance, uses a value as a parameter that you type in Datacap Studio, but you want it to read the value for the parameter from a database or `INI` file rather than have it hardcoded in your rule. In such a case, you can write an action that retrieves the value from wherever you want, and then calls the action that you need and passes in the value that you read in your script. When calling an action from your code, use the following functions:

- For old style actions that do not require a parameter:

  ```
  Call ActionName(false,false)
  ```

- For old style actions that do require a parameter:

  ```
  Call ActionName(false,false,parameter)
  ```

- For new style actions that do not require a parameter:

  ```
  Call ActionName()
  ```

- For new style actions requiring parameters:

  ```
  Call ActionName(comma separated list of values)
  ```

Note: You might need to specify an *include reference* when calling an action outside of your current library.

13.2.6 The include reference and its importance

The include reference is noted at the top of your script code in an XML `<i>` tag:

```
<rrx namespace="ScriptClass" v="8.0.0"><i ref="rrunner"/><i ref="invoice"/><<g>
```

It causes the action library listed to be loaded before your script is loaded, ensuring that the function that you want to call from the other library is loaded into memory when you call it.

Datacap loads the libraries as they are needed, so during the execution of a Task Profile, it might load several libraries that it detects are needed.

> **Note:** When the Script control tries to load a function from a library that has the same name as a function previously loaded, the function that is loaded last overwrites the function that was loaded previously.

Loading functions with the same name but from different libraries can create an interesting effect. For example, MyAction is called and run from one library. If another library containing an action also called MyAction is loaded later during the profile, different code (the code from the new library) will run. If you are using an action with the same name as some other action in a library, you need to know which one has been loaded last at any specified time.

You can control this with the include reference. For example, assume that you want to improve on an action that you find in a ready-for-use library. You can copy the action to your own library, but you must include a reference to the ready-for-use library to ensure that your modified code always overwrites the ready-for-use code. This way, you can use all of the actions from the standard library that you do not feel need modification and overwrite only the action or actions from the library that you do want to modify. This still enables you to take advantage of enhancements or additions of other actions to the standard library in future releases or patches, yet overwrite only the one action from the library with your own code.

13.2.7 Language choice

Many customers prefer their scripts written in VBScript because the script files can reside on a network directory, ensuring all users run the same code. It is also viewed as simpler and easier to update than compiled C# code.

With Datacap 9, the resultant .NET `.dll` file built from the new .NET 4.0 libraries no longer needs to be registered on individual machines. This overcomes one of the major hurdles for using C# in the past. There is more configuration, but the libraries can be accessed and shared from a network directory.

As a result of the changes in Datacap 9, deciding which scripting language to use depends largely on your preference for and comfort with each language and on your willingness to maintain the source files in case the script library needs enhancement in the future.

13.3 Getting started

The template for writing an action library in C# is included in the IBM Datacap Developer Kit.

1. Download the kit from IBM developerWorks:

   http://ibm.co/1MgEoI4

2. The VBScript action library template can also be downloaded from the Datacap Technical Mastery Community section on developerWorks. In the Files section, download `ScriptClass.rrx`.

3. We show the examples in this chapter with VBScript. After `ScriptClass.rrx` is downloaded, save it in a place where you store your important files, make a copy of it, and put it in the `/rules` folder in your application directory:

   `\datacap\appname\dco_appname\rules`

4. Next, change the name of the file to your desired name, but leave the `RRX` extension intact.

5. Open the file in a text editor and change the namespace attribute, on the second line, to the same root name that you gave your file:

   ```
   <rrx namespace="MyFileName" v="8.0.0"><i ref="rrunner"/><g>
   ```

   You might also want to add `<i>` tags to this line if you are writing actions as described earlier in this chapter. This loads other action libraries before the actions in this library are loaded. That way, you can overwrite actions from other libraries that are currently held in memory, or you can call their functions or actions from your own actions.

13.4 Documenting your action

In VBScript, lines that start with an apostrophe are comments, and any text that follows is ignored. Comment your actions liberally. A block of comments at the start of the `ScriptClass.rrx` file is used for versioning, as shown in Example 13-5.

Example 13-5 Block comment

```
'*********************************
'Scripting Class Actions
'ScriptClass.rrx
'IBM Corporation (c)2011

' Version
'  8.0.0 - 02/14/2011   Tom Stuart
'        - Original Scripting Class RRX File
'
'*********************************
```

Each action is also documented with a series of XML tags (as shown in Example 13-6). These tags are read when you right-click the action from your library in Datacap Studio.

Example 13-6 XML tags

```
<ap>
   Document your parameters here<br/>
  </ap>
  <h>
         Explain the use of your action here<br/>
    <e>
        Place an example action call<br/>
    </e>
  </h>
  <lvl>
      Place the level (Batch,Document,Page, or Field) that you need the action to
be run from if there
      is any dependency.
  </lvl>
  <ret>
    List your return conditions. <br/>
    <b>TRUE or FALSE</b> If you want to affect the rules order execution for some
reason, or if this is a validation
    action, you might want to conditionally return <b>FALSE.</b>  Otherwise,
<b>TRUE.</b>
  </ret>
  <see>
```

```
    Reference other related actions here <br/>
    <scr>RelatedFunctionName</scr>
  </see>
  <p name="Parameter1"/>
  <p name="Parameter2" />
```

The tags in this example produce the output shown in Figure 13-4.

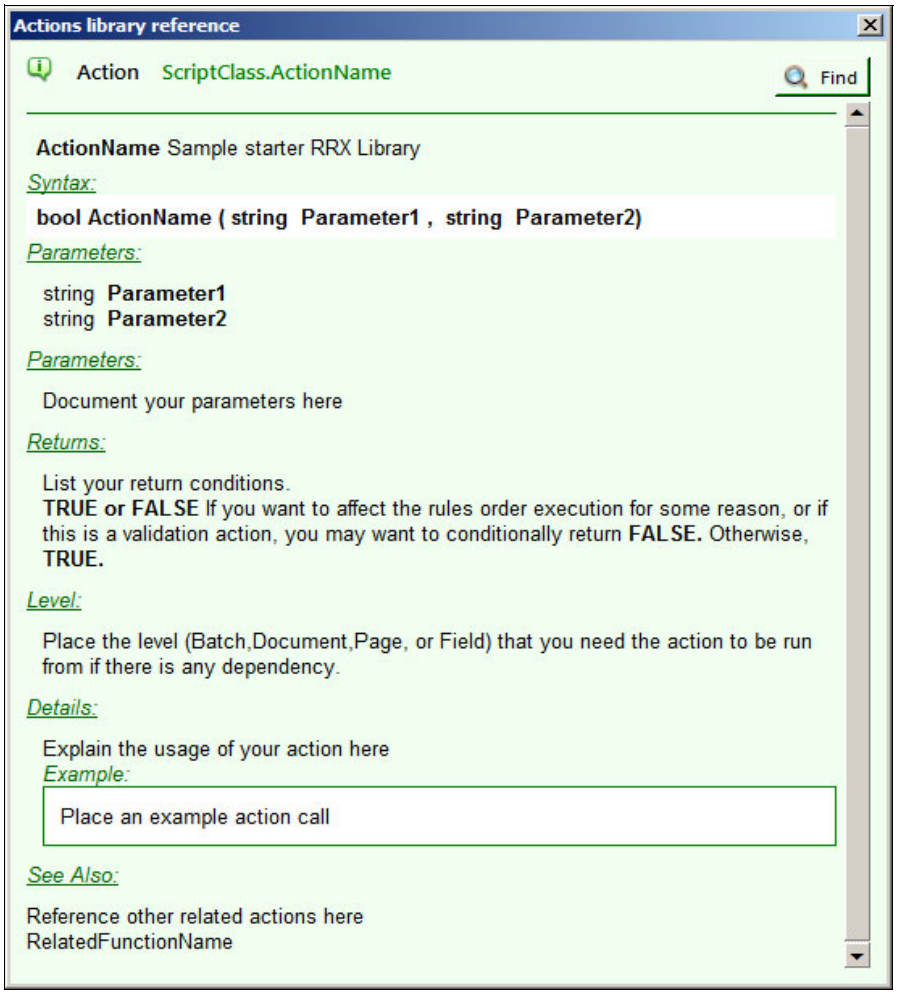


Figure 13-4 Output of the default action documentation tags in ScriptClass.rrx

These tags are written to be self-explanatory in `ScriptClass.rrx`, and each of them is optional. If you want to see how your action documentation is displayed, just save it and right-click your action in Datacap Studio. You will see your results instantly.

The `<p>` tags at the bottom of this section of code are parameters that you expect people using your action to supply when they use the action. You can add or subtract them to match the number of parameters in your action.

It is possible to type the data in these lines, but older versions of Datacap Studio might not enforce any data typing you set. Indicate the data type as this example shows:

```
<p name="dpi" type="int" />
```

13.5 Writing an action

To write an action, you put into practice everything covered in this chapter so far.

13.5.1 Writelog

An essential tool in software development is the ability to log what is happening. *Writelog* is a function in the `rrunner` action library that outputs a line of text to the log file, which is useful for debugging your actions. Adding `WriteLog` in various places enables you to “see” what is happening, as Example 13-7 illustrates:

Example 13-7 Adding a log statement to your action code

```
For i= 1 to 3
   Writelog("This line will be output to the log." & cStr(i) )
Next 'i
```

13.5.2 The CurrentObj and DCO objects

When you have written a new action, you set the action to run on an object in the document hierarchy, or DCO, such as a batch, document, page, or field. In many cases, when you attach an action to a DCO object, you want to manipulate to the object that you attached it to in some way. To do this, use the predefined object called *CurrentObj*, which references the object that your script is bound to. For instance, if you bind your object to a field, the following code writes the text value of that field to the log:

```
Writelog("My field value is: " & CurrentObj.Text)
```

DCO is another predefined object reference that always is the topmost (batch) object of the DCO:

```
Writelog("My batch name is: " & DCO.ID)
```

The methods and properties of CurrentObj, DCO, and any other document hierarchy object that you can reference from your script (namely the batch and every document, page, field, and character in the batch) are documented in the Datacap documentation section of the IBM Knowledge Center:

http://ibm.co/1GDLuiX

13.5.3 A word about variables

Variables are useful. They are often used to track statistics or to store specific information in the Page File (for example, `verify.xml` in the batch folder) or the Data File (for example, `tm000001.xml`in the batch folder) to help with debugging. Most programmers use them liberally.

There are several DCO methods in the Datacap object API to read and write variables. These provide little error checking, so instead, use the properties (as opposed to methods) to get and set variables. It is easy and intuitive, as Example 13-8 shows.

Example 13-8 Setting variables in your action code

```
CurrentObj.Variable("MyVar") = "This is my value"
VarString = CurrentObj.Variable("MyVar")
```

See “Datacap object API reference” in the IBM Knowledge Center for more information:

http://ibm.co/1LjLAlb

13.5.4 ObjectType

`ObjectType` is a method that returns the base type of your object. It is not to be confused with the Type property, which is the generic name of your object. `ObjectType` returns one of five values, depending on whether your script is attached to a batch, document, page, field, or character object:

- 0 - Batch
- 1 - Document
- 2 - Page
- 3 - Field
- 4 - Character

You cannot attach actions to characters in the DCO. However, with scripting, it is possible to reach individual character objects to read or write their properties or call their methods. The value returned for `ObjectType` can tell you what type of document hierarchy object you are working with.

13.6 Referencing other objects from DCO or CurrentObj

Think of the DCO hierarchy of Batches, Documents, Pages, and Fields as Figure 13-5 shows.

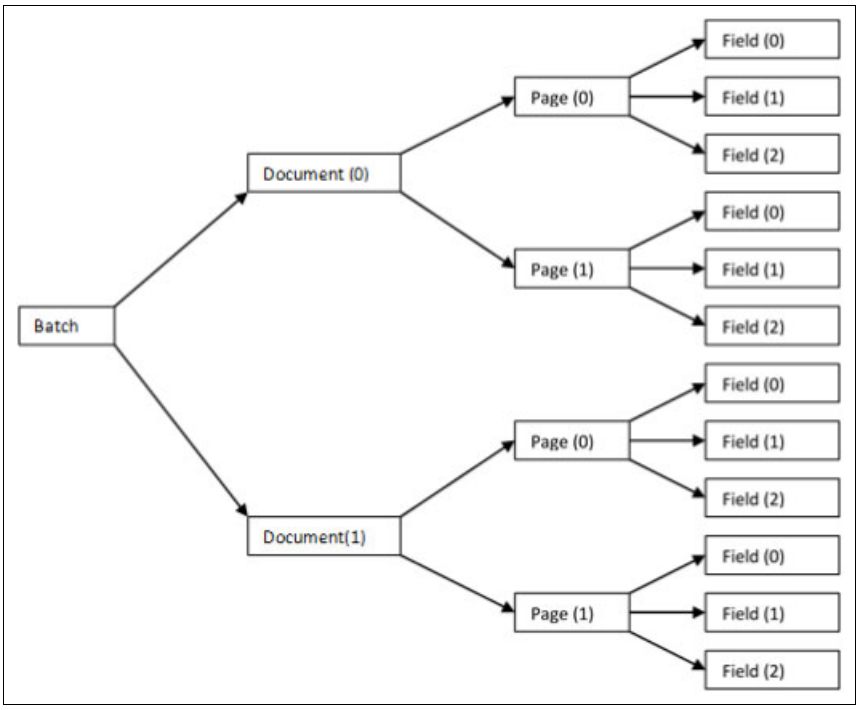


Figure 13-5 Graphical representation of the DCO objects

13.6.1 Finding the parent object

There are several ways you can reference other objects from the two base object references, CurrentObj and DCO.

Finding the parent object is easy. Each object (except for the batch object) has a parent object, so if you are on a page that is in a document. Then, the following code brings you to the document object from an action bound to the page:

```
Dim oDoc    'objects must be dimensioned in VBScript, like this
Set oDoc = CurrentObj.Parent
```

Similarly, `Set oDoc = CurrentObj.Parent.Parent` brings you to the document from a field object. The only thing to consider when writing such an action is that, sometimes, a page might not be part of a document or perhaps your field is a child of another field, which can occur with line items in APT. Perhaps someone wants to run your action on a page rather than a field. If you want a more powerful method of returning the document, do something similar to what Example 13-9 shows.

Example 13-9 Finding your document object

```
Dim oDoc
Set oDoc = CurrentObj         'sets oDoc to the bound object
While oDoc.ObjectType > 1     'test to see if oDoc is a document
   Set oDoc = oDoc.Parent     'move up to the parent object if not
Wend                          'ends the while loop
If oDoc.ObjectType = 0 then   'is oDoc pointing to the batch object?
   Writelog("Document Object was not found.  Exiting.")
   Exit Function
End if
```

At the end of this code, `oDoc` should point to your document object, and you can set variables, delete or add pages, set properties, or do whatever the DCO methods allow you to do.

13.6.2 Finding child objects

Finding child objects is a bit more difficult, depending on how you want to refer to the child object.

To find a child object by its ID, the method is `CurrentObj.FindChild(ChildID)`. This works, but remember that the ID is the unique identifier of each object. Your application might have a document that contains a Page1 and a Page2, but from the document, referencing Page1 would need the ID, which is probably something like TM00001 (the unique identifier).

If you know the position of the child object in your document, for example, Page1 is always the first page of the document, so the better approach is to use `CurrentObj.GetChild(index)`. It is zero-based, so `CurrentObj.GetChild(0)` will always give you the first page on the document.

If you have a situation where the order might change, you can search for a type by looping `GetChild`. For example, if your script was bound to a document, and you want to find a page called *SignaturePage* that is not assured to be at a certain index, you could do as Example 13-10 shows.

Example 13-10 Searching for a type by looping GetChild

```
Dim oSigPage
For I = 0 to CurrentObj.NumOfChildren-1  'loop through all children
   If CurrentObj.GetChild(i).Type = "SignaturePage" then
      Set oSigPage = CurrentObj.GetChild(i)
   End if
Next 'i
```

At this point in your code, ***oSigPage*** points to a page with the SignaturePage type. Normally you would write such an action with the actual value that you are looking for, expressed as a parameter to your action to make it more usable.

One last thing to notice: In the code in Example 13-11, we pass in the field name as a parameter. However, if the parameter value is wrong or the field does not exist, it fails to find it. Because we set `oField` to `Nothing` at the beginning, we can check whether it still is set to `Nothing` and exit (that is, return `false`).

Example 13-11 Checking whether oField is still is set to Nothing

```
Dim oField
Set oField = Nothing
Set oField = CurrentObj.FindChild(MyParameterValue)
If oField Is Nothing then
    Writelog("Field does not exist.  Exiting")
   Exit Function
End if
```

If your field is found, you can be sure that `oField` is, in fact, a field on the page, and you can safely set properties or run methods on it.

14

Classification and separation

This chapter describes how Datacap classifies pages and documents and separated the incoming stream of pages into separate documents. It describes the standard rulesets included in Datacap that implement these capabilities.

This chapter includes the following topics:

- Overview
- Classification process
- Classification using the Identify Pages ruleset
- Creating documents
- Document integrity

14.1 Overview

When documents are captured, they are often scanned or imported in a single batch of multiple documents. There might be many pages, and often no indication where the documents begin and end. The batch is a stream of pages, and it is up to Datacap to apply structure to it. This includes determining what types of documents and pages are in the batch and creating the documents; in other words classification and separation.

Classification is the process of identifying document and page types. Classification assigns a page type to each page. With Datacap, document-level classification is determined based on the page types. By identifying the page types, classification also provides the information needed to separate documents. *Separation* creates documents by determining the starting page of each document. Certain page types are flagged as the first page of a document. So, a new document begins with one of these page types.

Classification is used in several ways when documents are processed. Classification is the basis for determining where each document begins and ends. Classification also determines what data is associated with or collected for a document or page. The Datacap rules run conditionally based on the page and document types. Therefore, effective and accurate classification is important.

Figure 14-1 illustrates the concept of the classification and separation.

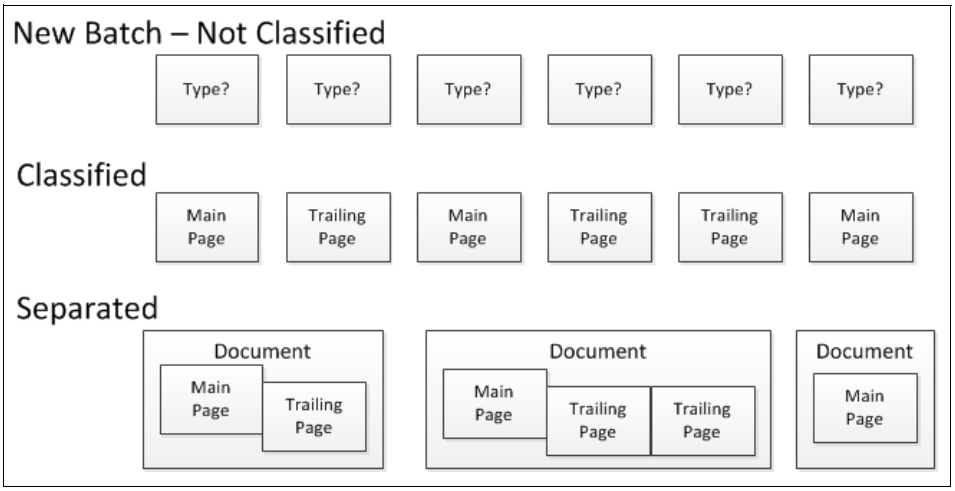


Figure 14-1 Classification and separation

Datacap supports a variety of classification methods. The methods that you use depend on your documents and how you structure your batches. The variety of classification methods give you the flexibility to deal with the variability that occurs with printed documents. You can use a single method or use multiple methods in combination. The Datacap user interfaces also provide ways for users to manually classify pages and documents. Classification is primarily done by the Identify Pages ruleset which supports the following classification methods:

- Blank page detection
- Page source location
- Bar code recognition
- Fingerprint matching
- Locate using keyword
- Content classification
- Last page type

You can also create your own classification rulesets implementing custom rules or extending the capabilities of the Identify Pages ruleset. In our examples we show both approaches. The Marketing Postcard application uses the Identify Pages ruleset; the Bank Statement application uses custom rules.

The Marketing Postcard example in Chapter 6, “Structured forms application” on page 131 uses the Identify Pages ruleset. It has a Postcard document type that includes the Campaign1 page type. The system uses fingerprinting to classify or identify the page type. The page type in turn determines the specific data fields on the page and in the document. In this case, the Campaign1 page type defines the data fields on the postcard. The settings on the page type determine that each Campaign1 page is a separate single-page document. The rules that link to the Campaign page then specify that the data will be read from the page using field-level intelligent character recognition (ICR). If different marketing campaigns needed different data, new page types can be created to identify the new variations of the Marketing Postcard.

The Bank Statements example in Chapter 7, “Unstructured document application” on page 147 uses a custom PageID ruleset. It uses bar codes and rules for classification. In this example, the document types are Document and Separator. The Document type can include several pages:

- Main_Page
- Trailing_Page
- Attachment_Separator
- Attachment

The Separator type includes only the Separator_Sheet. The system uses bar codes to identify the Separator_Sheet and Attchment_Separator. Main_Page, Trailing_Page, and Attachment are identified using rules based on their position in the batch or the position relative to a Separator_Sheet or Attachement_Separator page.

The data is contained as fields on the Main_Page. The rules that link to the Main_Page and Trailing_page determines which pages are read full-page optical character recognition (OCR) and what fields are filled from the text of the OCR-scanned pages. In this case, fingerprinting is not used for classification, instead it only determines which template is used for Main_Page. If a new type of bank statement is received, then a new fingerprint is created, but the page type is still set to Main_Page.

These two examples show common approaches to classification however there are additional ways classify that are described in this chapter. The chapter focuses on using the Identify Pages ruleset.

14.2 Classification process

Classification and separation are done as a sequence of tasks in a task profile, such as PageID or Profiler.

Figure 14-2 shows these tasks within the Forms template applications using the PageID task profile. The tasks that implement classification and separation are Identify Pages, Create Documents, and Document Integrity.

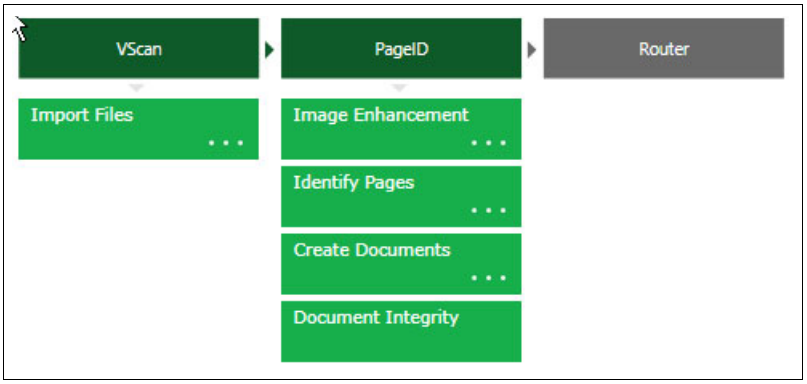


Figure 14-2 Classification tasks within the PageID Task Profile

Identify Pages classifies pages, assigning a page type to each page in the batch. When pages are added to a new batch, they are assigned the page type of *Other*. All of the page types are set this way so that you can identify pages that have not been classified. When PageID is completed, pages that have been identified are set to the page types that you define in the Batch Structure. Pages that are not identified remain set as type Other.

After pages are classified, documents can be created. Create Documents groups the pages into documents, assigns types to the documents, and creates fields on the batch, documents, and pages. Document Integrity checks the results and flags any problems so that they can be reviewed and corrected by a user. Documents and pages that have problems have their status set to a value of **1** (one).

14.3 Classification using the Identify Pages ruleset

Now we describe the Identify Pages ruleset and the classification methods that it includes. Figure 14-3 shows the sections of settings. We cover each of these sections in this chapter.

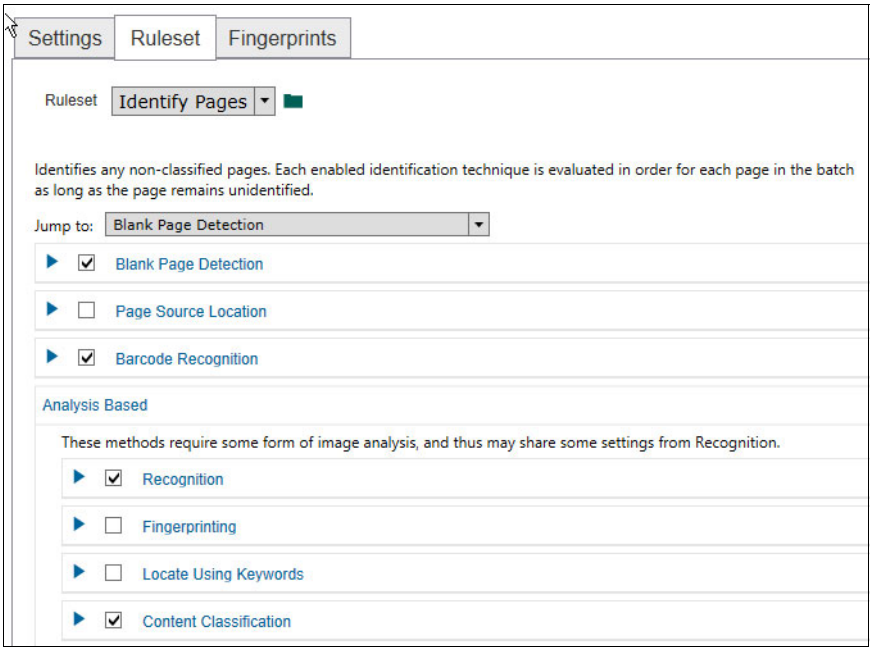


Figure 14-3 Identify Pages ruleset sections

The Identify Pages ruleset is configured on a single panel at the batch level. It is usually run after Image Enhancement (and if used, after Convert Files to Images). Note that you can run additional image enhancement later because the Enhance Image ruleset can vary its settings based on the page type.

In Identify Pages, when a batch is created, because the types of the pages are undetermined, all of the pages have the default type of Other. Starting with the first page of the batch, Identify Pages tries to set each page to a new page type by going through each method. When one of the methods succeeds, the page type is set. The order of the methods listed on the ruleset panel is the order the system runs them. If none of the methods identify a page, the page type remains set to Other. This continues with next page and so forth through the entire batch. When this process is complete, each page is either set to a new page type, or it is still set as Other.

The Identify Pages ruleset supports most of the common classification scenarios. However, you might encounter situations that require other methods or a different processing sequence. In this case, you can create your own rulesets in Datacap Studio. You can use Identify Pages and add a ruleset before or after it. You also can configure the Identify Pages ruleset, make a copy of the ruleset, then edit the copy. In this way, you can implement your own methods and combine them with the standard Datacap classification methods.

14.3.1 Blank page detection

The Blank Page Detection section (Figure 14-4) configures the Identify Pages ruleset to detect pages that have little or no content on them. For example, a back side of a sheet of paper that is scanned is blank. This method flags blank pages by assigning a page type. You can define a specific page type for this purpose. Blank pages are not deleted by this ruleset, it only sets their page type. If you want to delete blank pages, you need to include additional rules in your application using Datacap Studio.

The system checks the size of the page. If it is less than the set Maximum size (bytes), the page type is changed to the type that you select in the Blank page type setting.

You can also have the system check all, odd or even pages. The odd or even setting can be used when scanning both the front and pack side of pages where the back side is often blank.

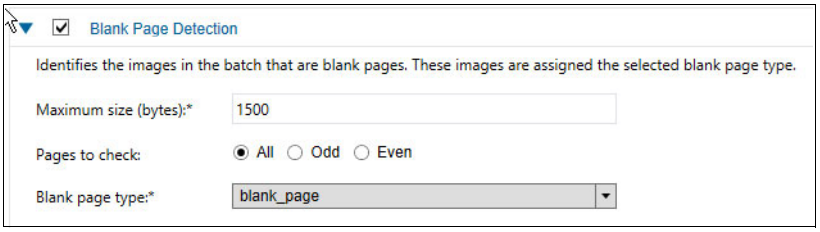


Figure 14-4 Blank Page Detection section

14.3.2 Page source location

This method is used with electronic documents such as emails, PDF, multi-page TIFF files, Microsoft Word documents, and Microsoft Excel documents. In many cases, each of these files is already a separate document. You can use the Convert Files to Images ruleset to convert these electronic documents to single TIFF pages, and then use the page source location to classify the pages as main and trailing page types.

The system sets the first page of a file to the main page type and sets all of the remaining pages to the trailing page type. In Figure 14-5, the first page of a document is set to Main_Page, and the subsequent pages are set to Trailing_Page.

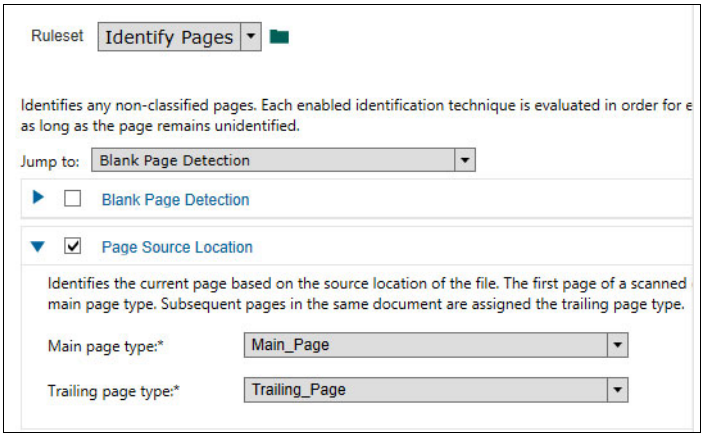


Figure 14-5 Identify Pages ruleset

14.3.3 Bar code recognition

This method is used when a page contains a bar code that identifies the document or page type. You define a list of the bar codes and their corresponding page types. The system looks for bar codes on the pages and if one is found, the page type is set. You can use this with pre-printed separator sheets or with bar codes that are printed on the documents.

The images must be black and white single page TIFF format. If you want to use this method on other file types, you can convert the pages using Convert Files to Images ruleset before using Identify Pages.

For each page, the system reads all of the bar codes on the page that match the specified bar code types and confidence level. If one of the bar codes is in found in the Mappings list, then the page type is set to the corresponding Mapping Page Type. In the example in Figure 14-6, the UNKNOWN type is selected. This means that the system will check for any supported type of bar code. If a bar code is read with the value of Form1437, the page type is set to HCFA Form.

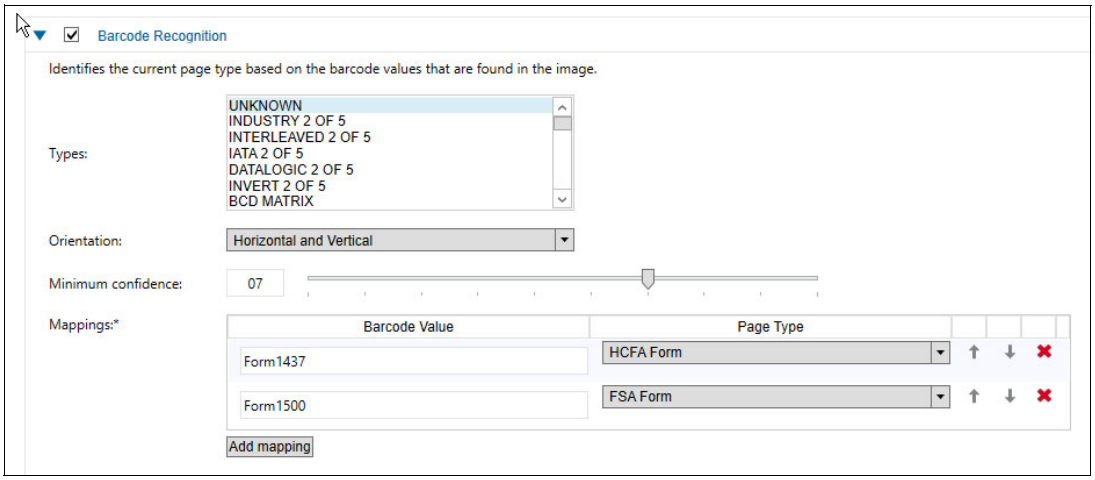


Figure 14-6 Datacap Desktop bar code recognition page

When you have bar codes that rely on vertical or horizontal lines, you must be careful. If you run Enhance Images before running Identify Pages, do not erase bar code lines by using the Remove Lines option. Make sure that Minimum length setting is shorter than the length of the bar code lines (Figure 14-7).

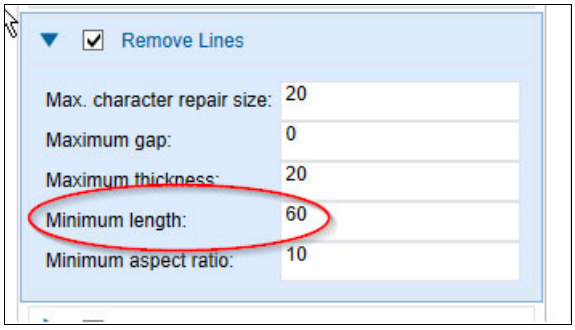


Figure 14-7 Remove Lines option

14.3.4 Analysis based settings

All of the classification methods that require analyzing content on a page are grouped together in the Analysis Based settings. This includes fingerprint recognition, locating keywords, and content classification. All of the methods share some common recognition settings so that the Recognition section is listed first followed by each of the classification methods.

Recognition settings

The settings in the Recognition section (Figure 14-8) cause the system to run full page recognition using the OCR/S engine. You can also set the system to detect and rotate images that are upside-down or sideways. If you do not need to read data from the entire page, you can speed up recognition by selecting a smaller area. The Recognition area setting defines the area of the page that will be read by the OCR engine.

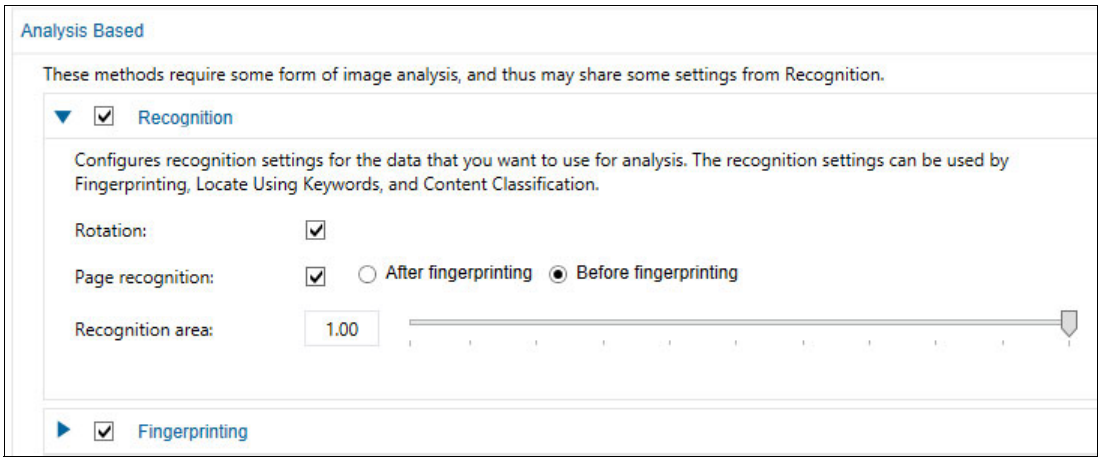


Figure 14-8 Recognition settings

14.3.5 Fingerprint recognition

Fingerprint recognition looks at the content of a page as a pattern similar to the swirls of a fingerprint on your finger. Datacap compares the pattern of the current image to a pattern database. This method is effective with pages that have a consistent appearance and so it is used with forms. This method is not suggested for unstructured documents such as letters of correspondence.

If there is a match that exceeds the specified confidence level, then the page type of the current image is set to the type of the matching fingerprint. If there is more than one match, the system selects the matching fingerprint with the highest confidence.

The images must be black and white single page TIFF format. If you want to use this method on other file types, you can convert the pages using the Convert Files to Images ruleset before using Identify Pages.

There are several ways to add fingerprints. You can add fingerprints at design-time using the FastDoc (admin) on the Fingerprint tab or in Datacap Studio using the Zones tab. Fingerprints can also be added at run time. This is done in the Learning template and the Accounts Payable application.

Fingerprints do more than classify pages. Fingerprints are also the basis for zonal recognition. Each fingerprint also contains the zone of the data fields. So in addition to providing a page type, the match also identifies the location of the fields on the page. There can be multiple layouts of the same page type. For example, a new version of a form might have the same fields that put them in different positions. In this case, a new fingerprint defines the zone positions of the new form, and the old fingerprint continues to work with the older form.

With paper scanning, each page can be positioned or fed differently through the scanner. As a result, the field positions on a page are not in the same position on each scanned page. The matching algorithm accounts for this and calculates the offset between the scanned page and the matching fingerprint. The offset adjusts the location of the zones to match the slight shifts in position. This makes the data extraction more accurate.

Fingerprints must be specific to the resolution of the image. If you plan to scan in multiple resolutions, you need fingerprints for each resolution. If you have some images at 200 DPI and others at 300 DPI, then you need separate fingerprints for 200 DPI and 300 DPI.

To use fingerprints effectively, you need to understand how fingerprints are created. There are two ways to create fingerprints. One way is with full-page OCR recognition and the other way is with Analyze Image. For best results, you should have the same method create the fingerprints at design time and find them at run time.

Full page recognition creates fingerprints based on the text generated by the OCR engine using the patterns of text and white space on the page. Analyze Image creates fingerprints based on the bitmap image by using the patterns of light and dark areas of the page. For best results, do not mix the methods. If you do mix the methods, fingerprint matching and zone positioning will be less accurate.

The Forms template, used in this book with the Marketing Postcard application, uses Analyze Image fingerprints. The Learning template, used in this book with the Bank Statements application, uses Recognition fingerprints. You can change the method by editing the Add Fingerprint ruleset using Datacap Studio.

Some of the settings in the Recognition section (Figure 14-8 on page 316) must be updated depending on the type of fingerprints used in your application.

If you use OCR recognition-based fingerprints, select **Recognition**, then select the **Before fingerprinting** option (Figure 14-8 on page 316).

If you use Analyze Image fingerprints, you have two options:

- Option 1. Clear **Recognition**.

  Use this if you will not be using Keyword or Text Analysis to identify pages. In this case, you will be running recognition in a recognition separate ruleset such as “Populate Fields Using Keywords”.

- Option 2. Click **Recognition** and select the **After fingerprinting** option.

  Use this if you will be using Keyword or Text Analysis methods and used Analyze Image. In this case, because recognition is completed here, a subsequent recognition ruleset does not need to run recognition.

Fingerprint section

Let us look at the options that implement Fingerprint matching within the Identify Pages ruleset. In this section, you configure the primary fingerprint options. You can change the area of the page that is analyzed, set a confidence threshold, configure learning, and set the connection to a Fingerprint Service server.

Fingerprint folder

The “Fingerprint folder” setting (Figure 14-9) is for display only. This setting can be changed in the Datacap Application Manager utility.

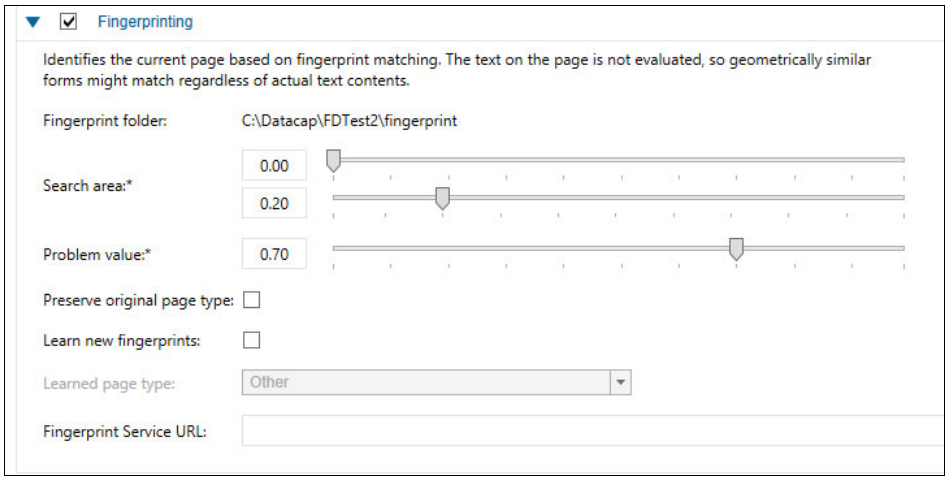


Figure 14-9 Fingerprinting folder settings

Search area

The “Search area” settings define the region of the page that will be compared to the fingerprint in the database. Zero is the top of the page and 1.00 is the bottom of the page. The first slider bar sets the vertical search area staring point. The second slider bar sets the vertical search area finishing point. In Figure 14-9, the first slider is 0 and the second slider is 20, so this sets the search area to the top 20% of the page.

The region is usually set to some portion of the top of the page because it is common for documents to vary in their layout in the top portion of forms. However you might need to try various settings to find the most effective range for your documents. A common practice is to use only a portion of the top of the page. Sometimes it is more effective to skip over the most top and start at 5%. This can skip over noise at the top of the page and fax scan lines that are also at the top.

Problem value

The “Problem value” setting defines the minimum confidence required for matching. In the setting above a value of .7 indicates that a match must be at least 70% confidence. Lowering the value might increase false positives, in other words, if the confidence is too low, the system might match to incorrect fingerprints more frequently. This level is an implementation decision. In your implementation, much confidence levels above .9 might be needed to achieve high accuracy. You might also need to create multiple fingerprints to provide more potential matches for each type of page.

Preserve original page type

The “Preserve original page type” setting can be used in special situations. If this setting is checked, then the page type is not changed but the other aspects of fingerprint can occur. When a fingerprint match occurs, three things can happen:

- The page type is set.
- Variables are set on the page (including *CCOFILE*, *TemplateID*, *Image_Offset*, and *Confidence*).
- Optionally, a new fingerprint is cataloged in the fingerprint database.

You might want to use this setting if you want to use the other aspects of fingerprint matching but leaves the Page Type unchanged.

Learn new page type

The “Learn new page type” setting is used to add new fingerprints at run time to the fingerprint database. If a new image does not match to a fingerprint, then this setting will save and catalog the fingerprint of the new image in the fingerprint database and folder. The “Learned page type setting” identifies the page type initially assigned to this new fingerprint. This is the first step in the process of learning new fingerprints. Learning is currently implemented in the Learning template.

Fingerprint Service URL

The “Fingerprint Service URL” sets the connection to a server or server farm running the Fingerprint Service. The fingerprint service is not required but it provides higher performance for systems that have many fingerprints. If you are using the Fingerprint Service, you should also use the Fingerprint Maintenance Tool, which is described in the online product documentation:

Maintaining fingerprints by using the Fingerprint Maintenance Tool

http://ibm.co/1UPOGC7

The current method of representing Fingerprints is through an XML descriptor called FPXML. The preferable practice is to use FPXML on all applications. For compatibility with an earlier version, the prior method of storing fingerprints is supported. However, we encourage you to use to the newer method. The templates and compiled rulesets in Datacap release 9.0 use FPXML.

If you match the page type using other classification methods and want to use zonal OCR or OMR, additional configuration is required in Datacap Studio. For example, if you identify a page using a bar code, then the system will not find a fingerprint because the bar code is matched before fingerprints are matched. Similarly, if you match using Locate Using Keyword, you must have either failed to match a fingerprint or fingerprinting was not selected.

In these circumstances, a ruleset can do the fingerprint match separately before the recognition rulesets. This can be done in either a custom ruleset that does both the OCR and Find Fingerprint operations. Then, either Recognize Pages and Fields or Populate Fields using keywords can populate the fields from zones.

Figure 14-10 shows an example of the rules are in a custom ruleset called Find Fingerprint. In this circumstance, the page type is already known, so the fingerprint search can use the SetFilter_PageType action to limit the fingerprint search to the known page type.

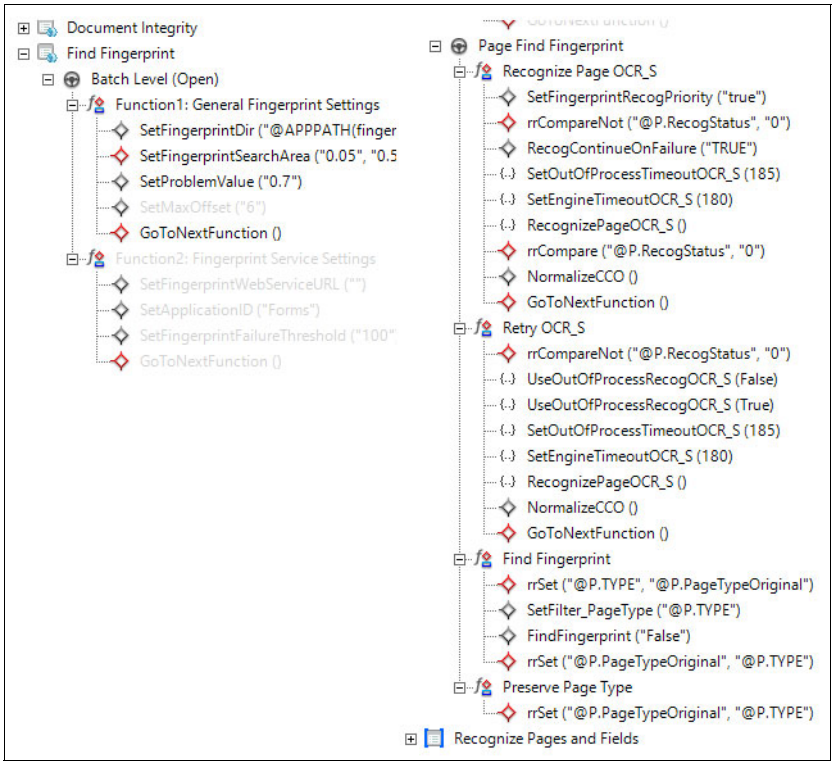


Figure 14-10 Find Fingerprint custom ruleset

14.3.6 Locate Using Keyword settings

Locate Using Keywords settings (Figure 14-11) identifies the current page based on keywords that are found in the recognized text.

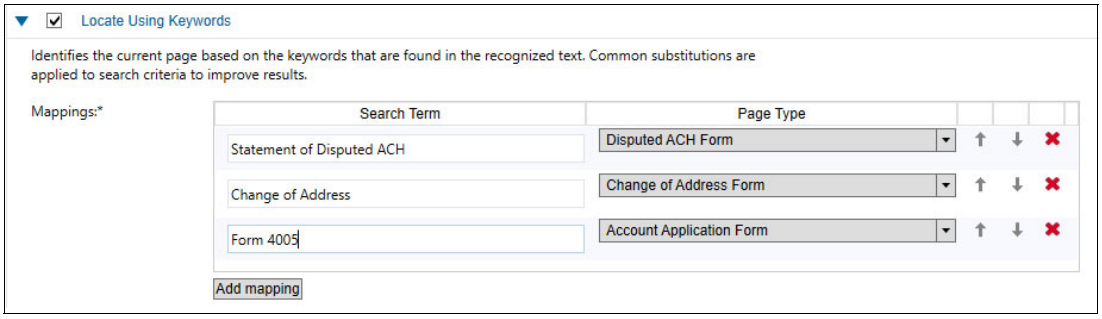


Figure 14-11 Locate Using Keywords setting

For example, in Figure 14-12, the document is titled “Statement of Disputed ACH Item.” Using this method, the page type is set when this text is found. In the settings shown in Figure 14-11 on page 320, the page type is set to “Disputed ACH Form.”

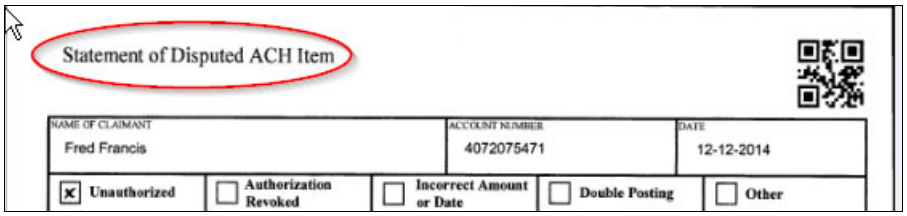


Figure 14-12 Sample document

The search looks for entire words so, you cannot truncate the search term in the middle of a word. For example. You can search for `Statement of Disputed`, but you cannot search for `tement of Dispu`. You must use the entire words. You can have more than one search term for each page. The system groups the search terms for a page type together. The search occurs from top to bottom, looking for the search terms for the first page type. If there is no match, the search continues with the second page type and so on until a match is found or all of the search terms have been checked.

To improve matching, the system automatically adjusts the search criteria to allow for common character substitutions. These are characters that can be misread but can still be reliably used if the search is broadened to account for them. Common substitutions include characters: B8, Z2, S5, oO0 and iItl1. For example, if the list includes the word “will,” the recognition engine might read the letters “i” and “l” as the number 1. So if the OCR results contain “w111,” it matches the word “will.”

There is a possibility of other errors in the OCR results besides the common substitutions. Therefore, consider the following recommendations to account for this:

- Larger text usually has fewer errors. Use any larger text words on the page that uniquely distinguish the page. Sometimes this text can be in the body of the page.
- Use the shortest unique word string. Extra characters make it more likely that errors in OCR could prevent the search term from matching.
- Use more than one search term for the same page type. If one search term fails, the other might still match. For example, you could include a search term from the page heading and another term from the page footing.

You can improve performance by following the following recommendations:

- Place the most frequent page types at the top of the list.
- If possible, search for text at the top of the pages.
- Limit the OCR region to less that the full page if possible. In some applications, you might not be using the full text for reading data fields on a page. Or, you might be able to use zonal recognition of only selected fields or pages in the recognition rulesets.

When you have many page types, it can be difficult to find unique search terms. In this case, you could consider using Content Classification or use a custom ruleset with additional search rules. For example, you could create a custom ruleset that searches for a second search term to distinguish pages that do not have a single unique search term.

14.3.7 Content Classification settings

The Content Classification method uses a statistical analysis of the text on a page compared to a knowledge base to find the closest match. This method uses the IBM Content Classification software product. Verify that your license with IBM entitles you to use this product. For more information about configuring it, see *IBM Content Classification documentation* in IBM Knowledge Center:

http://ibm.co/1Th8mkU

This method is similar to the Locate Using Keyword method in that it relies on the text from recognition. However, rather than defining specific keywords to search, it uses a knowledge base that is trained to classify pages based on text analysis. So, you do not need to enter specific search terms. Instead, you need to train a knowledge base with examples of each page type. With IBM Content Classification, Datacap passes a block of text to IBM Content Classification, which analyzes the text and compares it to its knowledge base of content. Different from keyword-based classification, IBM Content Classification uses linguistic analysis to determine the type of the page.

Figure 14-13 shows the settings:

- *Listener URL* sets the connection to the instance of Content Classification.
- *Language* sets the language that must correspond to the language in Content Classification.
- *Update Knowledge Base* sets the name of the Content Classification knowledge base.
- *Problem value* sets the minimum confidence threshold for matching a page to a Content Classification category.
- *Update Knowledge Base* indicates that the system should provide feedback to Content Classification when a match is found.

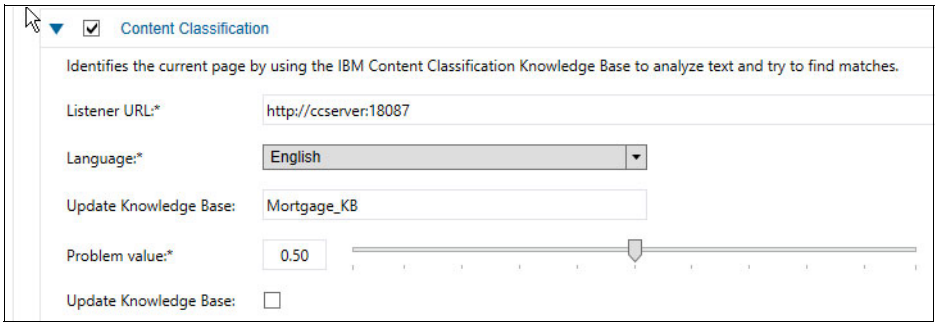


Figure 14-13 Content Classification settings

Before you use this method, you must create and do initial training of a knowledge base using Content Classification. This is done by using the Content Classification Workbench. Create categories in Content Classification that correspond to your page types in Datacap. Supply sample text for each category. Use a Datacap application to scan and export text files from your sample images. A minimum of 30 samples for each page type is suggested.

Over time, the knowledge base accuracy can increase by providing continuous feedback from your application. Although there is a setting for this purpose in the Identify Pages settings, it is not suggested to use that at this time. Provide feedback by adding additional rules to an export task using a custom ruleset.

Use the UpdateKnowledgeBaseCC action to assert that the classification was correct for each properly classified page. It is better to do this separately after any potential manual corrections by the Fixup or Verify tasks. If the feedback is done in the export task, then the training reflects these corrections and improves the knowledge base. Providing continuous feedback lets the knowledge base adjust to changes in the documents.

Figure 14-14 shows an example of knowledge base categories for mortgage documents in the IBM Content Classification Workbench (on the left) and the corresponding documents and pages in Datacap Studio. Notice that the categories correspond to the page types. A document type has a main page category and optionally a trailing page category. This allows the system to identify the starting page of a document separately from the rest of the pages of a document.

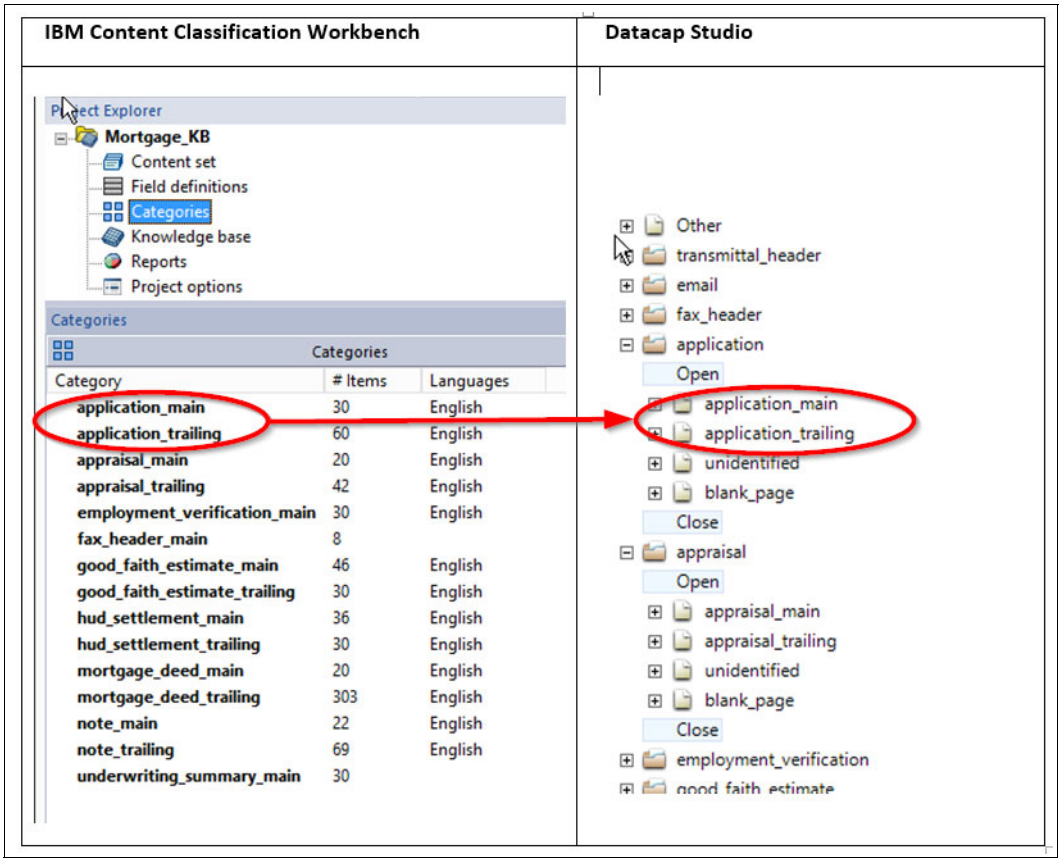


Figure 14-14 IBM Content Classification Workbench

This example also uses blank page identification. Each document type contains a blank_page type that allows a document to contain blank pages. Also, each document type contains a Unidentified page type. Any remaining Other page types are set to the Unidentified page type so that unidentified pages are assembled into the documents.

14.4 Creating documents

After your pages are classified, documents can be created using the Create Documents ruleset which groups pages together to form documents. It also sets the document type and creates the fields on the batch, documents, and pages. Internally, Datacap does this by updating the task XML and page XML files.

The Create Documents ruleset classifies and separates documents. It is run after Identify Pages as shown previously in Figure 14-2 on page 312.

To accomplish its tasks, Create Documents compares the page types identified in the batch to the Batch Structure (also called the Document Hierarchy, DCO). The Batch Structure specifies which pages are contained in each type of document and which pages signify the start of a document. The page settings—Minimum, Maximum, and Order—determine where a document begins and ends. These settings are sometimes called *document integrity rules*.

Figure 14-15 shows the settings for the Ruleset. There are settings that apply to special circumstances. If you want to replace existing structure (for example from electronic documents), you can remove the structure and rebuild the documents. If you want to retain the existing structure and only create fields for your classified types, you can create fields only. If you want to create batch fields, you can select the corresponding setting.

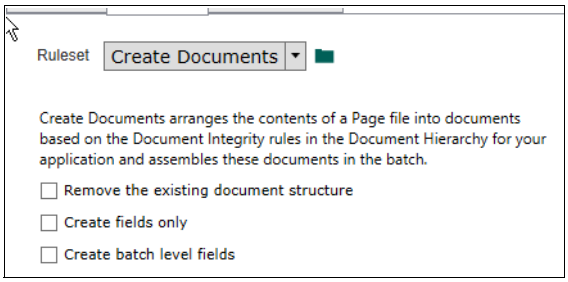


Figure 14-15 Ruleset settings

Figure 14-16 shows the settings for the first page of an application document. The first page is called *application_main*. Minimum and Maximum are both set to “1” to indicates that it is the beginning of a document. This allows for situations where a document type can have multiple formats with different starting pages.

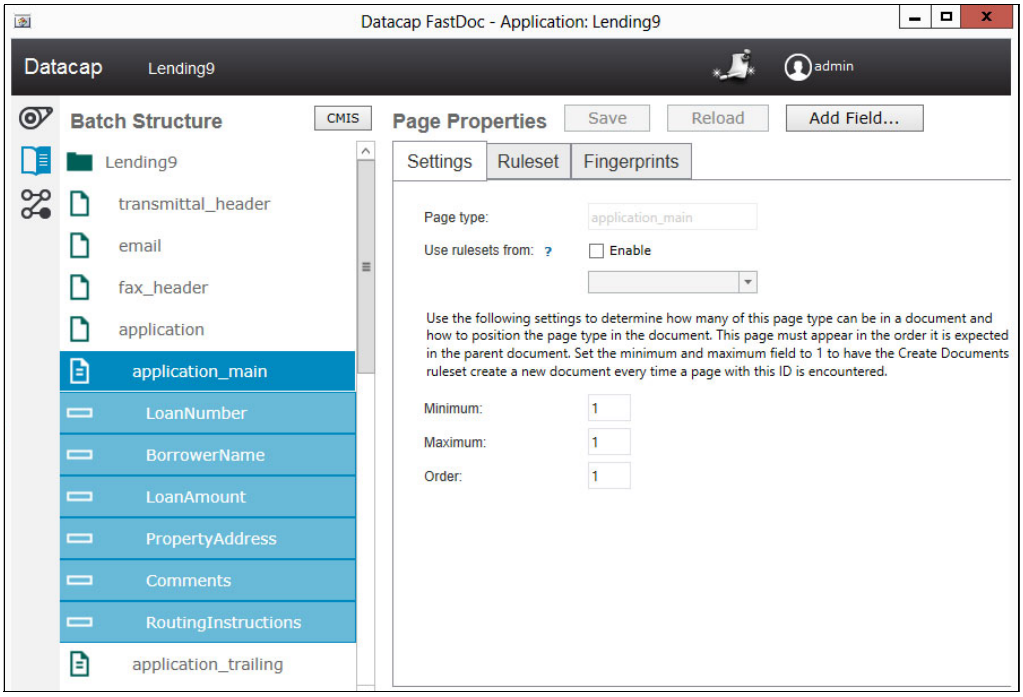


Figure 14-16 application_main page

The follow settings have these purposes:

- *Order* indicates the relative position of the page compared to other pages in the document (0 means any position). Because pages can be rearranged by users during Verify or Fix-up tasks, an order setting allows the document integrity checking to detect when users have placed pages out of order. Trailing pages have their order set to a number greater than the starting pages.
- *Minimum* sets the minimum number of pages of this type for each document (0 means no minimum).
- *Maximum* sets the maximum number of pages of this type for each document (0 means no maximum).

It is possible to have pages that are not contained in documents, but the most common scenario is to have pages within documents. You can also have a mix of pages and documents directly under the batch. If pages that are not contained in a document appear in the batch, make sure that you have included your page under a document and not directly under the batch.

After Create Documents is finished, pages and documents are identified and pages are grouped into distinct documents. If specified, fields are created on the batch, documents, pages and documents.

14.5 Document integrity

In the normal course of events, people make mistakes or might provide incomplete documents. For example, pages might be missing, out of order, or poorly scanned pages might be unreadable. Some way is needed to check that a batch is correct.

Document Integrity provides this check. Run this ruleset after Create Documents. There are many possible results that could be required intervention. For example, if a main page is not identified, you could have an Other page type followed by a list of trailing pages.

Document Integrity flags the batch, documents, and pages that fail to follow the Document Integrity settings. In the Marketing Postcard application, the workflow branches to a Fixup task so that a user can correct the batch manually. In the Bank Statements application, the process proceeds directly to data recognition and verification tasks.

The Document Integrity ruleset is not compiled and has no settings panel. If you need to make changes to it, you can use Datacap Studio.

Related publications

The publications listed in this section are particularly suitable for more information about the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information related to this book. Some publications referenced in this list might be available in softcopy only.

- *ACI Worldwide's BASE24-eps V6.2: A Supplement to SG24-7268*, REDP-4338
- *Disaster Recovery and Backup Solutions for IBM FileNet P8 Version 4.5.1 Systems*, SG24-7744
- *IBM FileNet Content Manager Implementation Best Practices and Recommendations*, SG24-7547
- *IBM FileNet P8 Platform and Architecture*, SG24-7667
- *IBM High Availability Solution for IBM FileNet P8 Systems*, SG24-7700
- *Introducing IBM FileNet Business Process Manager*, SG24-7509

You can search for, view, download, or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Online resources

These websites are also relevant as further information sources:

- IBM Datacap introduction

  http://www.ibm.com/software/products/en/datacap
- IBM Datacap: Mobile Document Capture Made Simple (2:36)

  https://www.youtube.com/watch?v=A6SYsUUamNw&feature=youtu.be
- IBM Datacap Version 9 documentation

  http://ibm.co/1GDLuiX
- IBM Document Imaging web page

  http://www.ibm.com/software/products/en/document-imaging-family
- IBM Production Imaging Edition web page

  http://www.ibm.com/software/products/en/prodimagedit
- IBM FileNet Content Manager web page

  http://www.ibm.com/software/products/en/filecontmana
- IBM Case Foundation web page

  http://www.ibm.com/software/products/en/case-foundation

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Redbooks

Implementing Document Imaging and Capture Solutions with IBM Datacap IBM Datacap

SG24-7969-01

ISBN 0738440906

IBM

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

SG24-7969-01

ISBN 0738440906

Printed in U.S.A.