

Forms Recognition Implementation Strategies for Large Enterprises

Introduction

Countless companies rely on paper forms for gathering information from customers, patients, students and the like. Automated forms recognition can be used on anything from a simple postcard to complex, multi-page tax forms to save time and money by increasing accuracy and reducing man-hours. However, large enterprises such as financial institutions, government agencies and hospitals often deal with huge numbers of forms on a daily basis, causing unique problems which can be successfully and efficiently handled with good planning, form design and the right software.

First, there are two different kinds of forms used within any forms recognition application: master and filled. Master forms are the blank templates that define where data is to be extracted from. After customers receive, complete and submit their filled forms, those forms are then compared against the master forms (i.e. forms recognition or classification) and then the data is extracted (i.e. forms processing). Large quantities of filled forms are a common within any organization, but the particular issue large enterprises can face is a multitude of master forms. Corporations with hundreds, even thousands of master forms must cope with an exponentially greater recognition time. Accuracy can degrade as well, since there may be similarities between forms causing false positives.

In this white paper we will discuss several strategies for dealing with large quantities of master forms, and how to leverage the LEADTOOLS Forms Recognition SDK to quickly and accurately process your forms. By implementing a combination of multi-threading, cloud computing, barcodes and two-phase categorized recognition, a forms recognition and processing application built with LEADTOOLS can handle any large-scale scenario you can throw at it.

Increasing Speed with Multiple Threads

In this day and age, using multiple threads to increase speed may go without saying. However, ensuring your application takes advantage of everything your hardware has to offer is a must. Most SDK vendors realize this and will tout that their libraries are “thread-safe,” but some consumers might not realize how vague that statement can be as it has no guarantees regarding what is going on under the hood. It is quite possible that the “thread-safe” function is classified as such simply because it forcibly runs on a single thread.

What sets LEADTOOLS apart from the pack is how simple, integrated and easy to control its multi-threading support is. You don't have to do the hard work of spawning threads, passing information and making sure it all runs without buffer overruns and memory leaks. When initializing the [AutoFormsEngine](#), simply pass an [IOcrEngine](#) for each processor core on your system and LEADTOOLS does the rest.

```
// Create an OCR Engine for each processor on the machine. This
// allows for optimal use of thread during recognition and processing.
ocrEngines = new List<IOcrEngine>();
for (int i = 0; i < Environment.ProcessorCount; i++)
{
    ocrEngines.Add(OcrEngineManager.CreateEngine(OcrEngineType.Advantage,
        false));
    ocrEngines[i].Startup(formsCodec, null, String.Empty, String.Empty);
}
// Point repository to directory with existing master forms
formsRepository = new DiskMasterFormsRepository(formsCodec,
    masterFormsFolder);
autoEngine = new AutoFormsEngine(formsRepository, ocrEngines, null,
    AutoFormsRecognitionManager.Default | AutoFormsRecognitionManager.Ocr,
    30, 80, true);
```

Distributed Processing with the Cloud

The LEADTOOLS Cloud SDK is based on the same underlying principles as multi-threading, but elevates it to the next level by using a network of computers to divide large workloads. By designing and controlling your own cloud infrastructure, there is virtually no limit to how far you can amp up the recognition speed.

The benefits of executing processor intensive tasks in the cloud go far beyond efficiency and speed. You can significantly cut your operating costs as well because any computer on your network can be used as a worker machine. Why purchase expensive dedicated servers when hundreds of desktop computers can pool their resources? LEADTOOLS provides ample customization to take advantage of unclaimed processing power on already in-use workstations. Is the employee engrossed in spreadsheets and word processors really getting his money's worth out of that eight core processor? You can choose the maximum CPU percentage, number of CPU cores, number of threads – even the time of day – that your worker process takes up on your employees' machines without hampering their regular duties.

The cloud is amazing for handling your abundant inflow of filled forms, but each node on your network can still be hampered by an abundance of master forms. Now that you have maximized the potential of your hardware, you must focus on intelligently designed master forms to continue getting more speed out of your enterprise-level forms recognition application.

Using Barcodes to Classify Forms

Barcodes are probably one of the fastest and most straight-forward methods of uniquely identifying one form from among hundreds of others. The most obvious benefit is that barcodes can pack a lot of information into a small space. This is especially the case for two-dimensional barcodes such as QR Codes, which are capable of storing up to 4,296 alphanumeric characters. Since these unique identifiers are small, they provide a very realistic means of up fitting a large fleet of in-production forms with minimal changes.

Why use forms recognition if you can identify the form with simple barcode recognition? At first glance, using forms recognition may seem like overkill but the advanced forms recognition and processing technology in LEADTOOLS comes with many benefits beyond classification. If you are planning on using OCR to recognize the user-provided information fields on the form, there are additional steps that must be taken to extract that information accurately. These include, but are not limited to: image clean-up, page alignment, compensating offsets for different DPI, and identifying the text to be extracted. Implementing these steps are a huge undertaking requiring thousands of lines of complex code, which LEADTOOLS does automatically with a fraction of the code.

```
// Set up the AutoFormsEngine to use Barcodes
autoEngine = new AutoFormsEngine(formsRepository, ocrEngines, null,
    AutoFormsRecognitionManager.Barcode, 30, 70, true);

// Run the forms recognition and processing on this document
AutoFormsRunResult runResult = autoEngine.Run(document, null);
if (runResult != null)
{
    // Process the recognized form and extract desired info
    foreach (FormPage formPage in runResult.FormFields)
    {
        foreach (FormField field in formPage)
        {
            // Do something with the extracted field data...
        }
    }
}
```

An Advanced Strategy: Two-Phase Recognition with Master Form Categories

For exceptionally large scenarios, it can actually be faster to run the forms recognition algorithm twice: once to identify the category and a second time to identify the form within the category. Time savings are realized by reducing the effective average number of master forms that the filled form must be compared against.

For example, let's say your company does business in multiple countries, states or regions. Each region uses similar forms but they have slight variations between them such as the contact information, logo or fields to gather. When designing the forms, identify and white-out the differences between the template images to create a category master form as shown below.

The figure illustrates the process of creating master form categories by removing differences between similar forms. The top row shows four original invoice forms (1, 2, 3, 4) with various differences. The bottom row shows two category master forms (A and B) where differences have been removed.

Form 1: Invoice form with a logo and contact information. It includes a table for shipping details and a table for items.

Ship To #	F.O. Number	Ship From	Ship Via	FOB	Terms

Quantity	Description	Unit Price	Total

Form 2: Similar to Form 1, but with a different logo and contact information.

Form 3: Invoice form with a different layout, including a table for shipping details and a table for items.

Ship To #	F.O. Number	Ship From	Ship Via	FOB	Terms

Quantity	Description	Unit Price	Total

Form 4: Similar to Form 3, but with a different logo and contact information.

Form A: Category master form with differences removed. It includes a table for shipping details and a table for items.

Ship To #	F.O. Number	Ship From	Ship Via	FOB	Terms

Quantity	Description	Unit Price	Total

Form B: Category master form with differences removed. It includes a table for shipping details and a table for items.

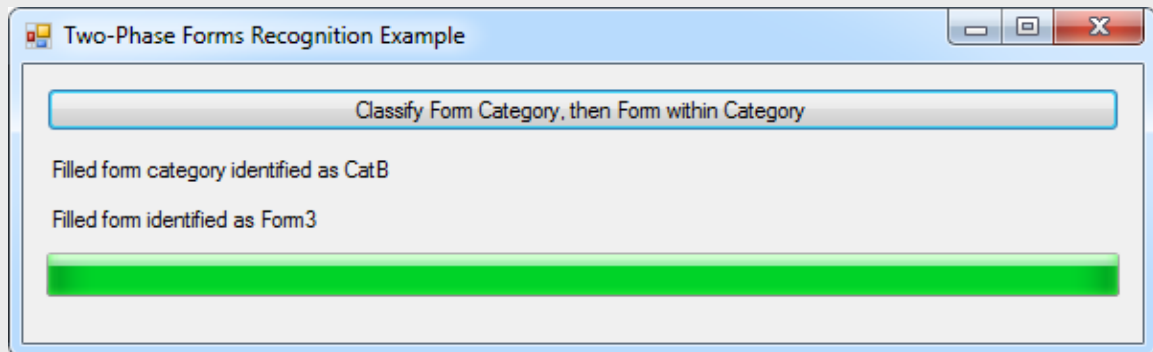
Ship To #	F.O. Number	Ship From	Ship Via	FOB	Terms

Quantity	Description	Unit Price	Total

Figure 1: Example of forms (top) with differences removed to create categories (bottom)

Looking at the examples above, you can see that if we run two-phase recognition on a filled Form 3, it will be identified as Category B and then compared against all of the other forms within Category B in the second pass when it is identified as Form 3. Using LEADTOOLS, you can narrow your forms recognition algorithm to only search specified categories within the repository. Though the two-phase recognition may seem like a complicated process, the code is quite simple:

```
// First pass we run recognition on the categories
recognizeResult = autoEngine.RecognizeForm(filledForm,
    formsRepository.RootCategory.ChildCategories.Where(
        i => i.Name == "Categories").ToList());
if (recognizeResult != null)
{
    // Second pass we run recognition on the forms within the found category
    foundMasterFormCategory = recognizeResult.MasterForm.Name;
    recognizeResult = autoEngine.RecognizeForm(filledForm,
        formsRepository.RootCategory.ChildCategories.Where(
            i => i.Name == foundMasterFormCategory).ToList());
    if (recognizeResult != null)
    {
        foundMasterForm = recognizeResult.MasterForm.Name;
    }
}
}
```



In the case of the simple example above with only two categories with two forms each, the speed benefits would be minimal. However, when implementing this strategy against a repository of hundreds or thousands of master forms, the optimizations should be immense. By combining intelligently designed forms with the award-winning document imaging technology within LEADTOOLS, you can create a forms recognition application that is perfectly matched for any large enterprise requiring a breakthrough in reducing time and costs.

Conclusion

This is just one of many real world solutions you can tackle with LEADTOOLS. Its state of the art Forms Recognition and Processing SDK is the most flexible and powerful product in its class, and LEADTOOLS offers an incredible value with its comprehensive family of toolkits for raster, document, medical and multimedia imaging. For more information on how LEAD Technologies can image-enable your application and boost your ROI, visit www.leadtools.com to download a free evaluation, or give us a call at +1-704-332-5532.

SALES: (704) 332-5532
SALES@LEADTOOLS.COM

SUPPORT: (704) 372-9681
SUPPORT@LEADTOOLS.COM



LEAD TECHNOLOGIES, INC.
1927 SOUTH TRYON STREET
SUITE 200
CHARLOTTE, NC 28203

About LEAD Technologies

With a rich history of over twenty years, LEAD has established itself as the world's leading provider of software development toolkits for document, medical, multimedia, raster and vector imaging. LEAD's flagship product, LEADTOOLS, holds the top position in every major country throughout the world and boasts a healthy, diverse customer base and strong list of corporate partners including some of the largest and most influential organizations from around the globe.

LEADTOOLS[®]
THE WORLD LEADER IN IMAGING SDKs

