Purpose: Document describes ClearCheck21 Image QA (IQA) API design principles.

Overview

ASC X9 Standards Committee on Banking is an Accredited Standards Committee (ASC) of American National Standards Institute (ANSI) is responsible for standardization in the area of Financial Services.

Working with Financial Services Technology Consortium (FSTC), X9 developed more formal set of test definitions outlined in the following sources:

- Test specifications are defined in "Check Image Test Registry" documents available [here](#).
- Reporting of test results is described in X9.100-180-2006 standard “Specifications for Electronic Exchange of Check and Image Data”. Specifically test reporting is described in Section “Image Test Detail Record (Type 56)”

Note that the Type 54 record, preserved for compatibility with old X9.37-2003 standard is not recommended for use in new applications. However, if needed, tests implemented in ClearCheck21 IQA can be used to set fields in the Type 54 record as well.

Inlite’s ClearCheck21 IQA is a software component that is packaged as a set of Windows COM objects for easy integration in various applications.
Scope of Metrics

It performs the measurements and reports the metrics and flags as required by the FSTC IQA tests listed in the X9 registry. ClearCheck21 IQA implements the subset of FSTC tests that applies to bitonal images and is the most significant for the usability of check images. The scope of the testing exceeds the IQA metric requirements mandated by the Federal Reserve in Sep 2007 for exchange images.

The specific metrics implemented by ClearCheck21 IQA are shown below:

<table>
<thead>
<tr>
<th>Metric/ Defect</th>
<th>X9 Registry Test Document</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undersize Image</td>
<td>Test_002</td>
<td>Image size in inches</td>
</tr>
<tr>
<td>Folded or Torn Document Corners</td>
<td>Test_003</td>
<td>Missing area in tenths of inches</td>
</tr>
<tr>
<td>Document Framing Error</td>
<td>Test_006</td>
<td>Extra scan area in tenths of inches</td>
</tr>
<tr>
<td>Excessive Document Skew</td>
<td>Test_013</td>
<td>Skew angle in degrees</td>
</tr>
<tr>
<td>Oversize Image</td>
<td>Test_001</td>
<td>Image size in inches</td>
</tr>
<tr>
<td>Image Too Light</td>
<td>Test_008</td>
<td>Pixel percentage</td>
</tr>
<tr>
<td>Image Too Dark</td>
<td>Test_007</td>
<td>Pixel percentage</td>
</tr>
<tr>
<td>Horizontal Streaks in Image</td>
<td>Test_015</td>
<td>Pixel percentage in scan line</td>
</tr>
<tr>
<td>Below Minimum Compressed Image Size</td>
<td>Test_011</td>
<td>Compressed image size in bytes</td>
</tr>
<tr>
<td>Above Maximum Compressed Image Size</td>
<td>Test_010</td>
<td>Compressed image size in bytes</td>
</tr>
<tr>
<td>Excessive “Spot Noise” in Image</td>
<td>Test_014</td>
<td>Count of noise spots</td>
</tr>
<tr>
<td>Front-Rear Image Dimension/Feature Mismatch</td>
<td>Test_009</td>
<td>Image size difference</td>
</tr>
</tbody>
</table>
ClearCheck21 IQA API

API reflects close relationships between ClearCheck21 IQA and X9 documents. The following demonstrates typical uses of API. Visual Basic 6 notation is use for simplicity,

Nomenclature

The nomenclature is represented as test/parameters/results. Sections are separated by “:”

- Definition source: If derived from the X9 registry, all names start with **X9**.
- The Test name is either:
  - Test X9 number and version separated by “-”: e.g. **003.00-00**
  OR
  - Test X9 XML name from registry: e.g. **FoldedTornDocCorners**
- Parameter name is either:
  - Parameter X9 Reference Number: e.g. **P1**
  OR
  - Parameter X9 XML name from registry: e.g. **MaxCornerFoldBRHThreshold**
- Result name is either:
  - Result X9 Reference Number: e.g. **R1**
  OR
  - Result X9 XML name from registry: e.g. **CornerFoldBRW**

Configuration

Configure the ClearCheck21 IQA process as follows. Note that the nomenclature offers several alternatives for convenience and clarity:

1. Create the IQA object:
   ```vba
   set oIQA = new ClearCheckIQA.IQA
   ```

2. To save processing time, you may disable tests that should NOT be performed:
   Using the test Number and Version
   ```vba
   oIQA.Front.Test(“X9:011.00-00”).Enabled = false
   ```
   OR using the XML name
   ```vba
   oIQA.Rear.Test(“X9:ImageOutOfFocus”).Enabled = false
   ```
3. Specify the parameter values of the test, for the front and for the rear image tests:
   Using the Test object:
   ```csharp
   oIQA.Front.Test("X9:UndersizeImageDimension").Parameter("P1").Value = 1234
   ```
   OR using the Test Number/Version/Parameter Reference:
   ```csharp
   oIQA.Front.Parameter("X9:003.00-00:P1").Value = 1234
   ```
   OR using the XML name:
   ```csharp
   oIQA.Rear.Parameter("X9:UndersizeImageDimension:MinImageHeightThreshold").Value = 1234
   ```
   NOTE: If any parameter for a test is not set, test is not executed. To test if parameter is set:
   ```csharp
   bIsSet = oIQA.Front.Parameter("X9:003.00-00:P1").isSet
   ```
   To assign default value to a parameter:
   ```csharp
   oIQA.Front.Parameter("X9:003.00-00:P1").SetDefault
   ```

   Check image processing

   Execute test and obtained results for each check:

   1. Assign images to front and back. For example check images is in 2-page TIFF file:
      ```csharp
      oIQA.Open("check.tif")
      ```
      To open front and rear images from arbitrary files
      ```csharp
      oIQA.Front.Image.Open("imageFront.tif", 1)
oIQA.Rear.Image.Open("imageRear.tif", 1)
      ```
   2. Execute test as configured:
      ```csharp
      oIQA.DoTests
      ```
   3. Retrieve and process front image results:
      ```csharp
      Dim oTest as CcTest
      ' Retrieve results for front image for each oTest in oIQA.Front.Tests
      ' Created application specific oType56 object
      if (oTest.isValid) then _
      CreateType56 (oTest.ImageTestFlag, _
      oTest.ImageTestNumber, oTest.ImageTestVersion, _
      oTest.TestResultsLength, oTest.TestResults, _
      oTest.TestParametersLength, oTest/TestParameters, _
      oTest.UserTestDataLength, oTest/UserTestData)
      next
      ```
      Note: CreateType56 method is customer application method that processes test results. For example it can prepare a Type 56 record to be added to a cash letter file.

   4. Retrieve and process rear image results. Use the same syntax as shown above, replacing Front with Rear.
5. To implement your own pass/fail logic, results of each test can be obtained directly. To check if test result value was measured:

```plaintext
isValid = _
oIQA.Front.Test("X9:UndersizeImageDimension:ImageHeight").isValid
```

To obtain value using test XML Name and Result Reference:
```
data = _
oIQA.Front.Test("X9:UndersizeImageDimension").Result("R1").Value
```

OR using Test Number/Version/Result Reference:
```
data = oIQA.Front.Result("X9:003.00-00:R1").Value
```

OR using Test Number/Version/XML Name:
```
data = oIQA.Front.Result("X9:003.00-00:ImageHeight").Value
```

OR using XML name:
```
data = _
oIQA.Rear.Result("X9:UndersizeImageDimension:ImageHeight").Value
```

**Error Reporting**

If errors are detected during test execution `DoTests` method return `false`, otherwise it returns `true`. Error information is reported in `ErrorText` properties. If no error is detect value of `ErrorText` property is an empty string.

```
If (oIQA.DoTests = false) then
    errorFront = oIQA.Front.ErrorText
    errorRear = oIQA.Rear.ErrorText
End if
```