Kofax Customer Communications Manager

5.0

Hammer3 Manual

for CCM Core

KOFAX
from Lexmark

Contents

Introduction

Hammer3 is a regression testing tool for ITP models and/or CCM Core setups. Hammer3 can be used to run a model on a (possibly remote) CCM Core and then compare the output of that model (actual output) to a base document (expected output). Hammer3 may also be configured to compare the output of two different CCM Cores against each other.

What's in a name

Why was this tool named Hammer3? Well, sometimes testing feels like banging software with a hammer to see where it breaks. As the reader may guess, there have been two previous versions which were used internally within Aia Software bv.

Hammer3 requirements

Hammer3 requires that Microsoft .Net Platform version 3.5 or higher is installed on your system. For the Image compare mode, it requires that your Microsoft Word installation on your CCM Core installation(s) is able to export documents as XPS.

When to use Hammer3

Hammer3 can be used in a couple of different scenarios:

- Changes to ITP code or content
- Changes to the infrastructure

In both of these scenarios the purpose of Hammer3 is to establish that no unexpected changes occur in the output of ITP models.

Using Hammer3 for changes to ITP templates or content

When implementing changes to reusable building blocks in ITP, certain aspects of the output are expected to change. Sometimes however, the changes made have unintended side effects to other templates. The CCM Repository may give an upfront impact analysis to establish which templates make use of the changed building block.

However, not every template that uses the changed building block will use the particular changes in all circumstances, e.g. depending on application data or user choice, some parts of a building block will not appear in an output document. With Hammer3, a regression test may be run to establish that only the expected changes are seen.

Hammer3 may also be run with specific test sets and data to test specific features of the document templates.

Using Hammer3 for changes to the infrastructure

Before implementing important changes to the ITP infrastructure, e.g. upgrading to a new version of the word processor, a new version of the operating system or a new version of the software, one may want to establish that such a change does not impact the document output.

In this situation, one may define a set of models and model documents together with the documents currently being output by the current system. The new system may then be setup. Hammer3 can then used to check and verify that the resulting output is identical to the output produced by the old system.

This usage of Hammer3 is not intended for validating specific features of document templates but only for establishing that the changes being made to the infrastructure have no unexpected side effects. Since one is not testing or looking for specific features it is good enough to test with a representative sample of the production output.

Installation

Hammer3 is shipped as part of ITP/Server from version 3.5.21.

After installing ITP/Server, you will find the Hammer3 executable and supporting DLLs in the Tools\Hammer3 subdirectory in the directory where you installed ITP/Server. You can run Hammer3 from this directory or copy the Hammer3 directory to any other location.

You should note however, that Hammer3 is intended to be run as a console application.

Invoking and configuring Hammer3

Basically Hammer3 is intended as a console application. Thus it is invoked from the command line:

```
C:/Test>hammer3.exe base.hammer3
```

where the argument given is the Hammer3 test specification file. Optionally you may add the option /v (verbose execution) to inspect the Hammer3 execution in more detail.

The Hammer3 test specification file

The Hammer3 test specification file is an XML document which starts with a configuration section, followed either by a number of test set sections, each referring to a set of tests to be run, or by a number of compose sections, each describing a particular test to be run.

A test specification file with only test sections then typically has the following form:

```
<?xml version="1.0" encoding="utf-8"?>
<hammer3 xmlns="http://www.aia-itp.com/Hammer3">
  <configuration>
          ...
  </configuration>
  <testset src="BaseTestSet.hammer3" />
  <testset src="Letters.hammer3" />
      ....
</hammer3>
```

A test specification file with particular tests may look like:

```
<?xml version="1.0" encoding="utf-8"?>
<hammer3 xmlns="http://www.aia-itp.com/Hammer3">
  <configuration>
          ...
  </configuration>
  <compose name="Begin-end end hash"
           type="Word"
           expecteddifferences="0">
          ...
  </compose>
  ....
</hammer3>
```

The configuration section

The configuration section contains a number of settings to configure Hammer3 for all following tests. Currently it may contain 6 parts, of which 3 parts are obligatory:

```
<configuration>
  <testdirectory name="C:\Test\Hammer3\Test\LetterTests" />
  <resultdirectory name="C:\Test\Hammer3\Test\Results" />
  <tempdirectory name="C:\Test\Hammer3\Test\Temp" />
  <keeptempfiles value="true" />
  <expected>
    <itpserver host="itpproduction"
               port="3001" />
  </expected>
  <actual>
    <itpserver host="192.168.126.128"
               port="3001" />
  </actual>
</configuration>
```

testdirectory
The test directory part specifies the location where all test sources are intended to be located. When a relative location is indicated, it is taken to be relative to the directory in which the test specification file is located.

resultdirectory
The result directory part specifies where the results of the Hammer3 run must be written to.

tempdirectory
The temp directory part is optional: it specifies the location where Hammer3 may write the intermediate results of the run, which may be useful while debugging. When this directory is not specified a system temporary directory is used.

keeptempfiles
The keeptempfiles part may be used to retain the intermediate results after the run. When the temp directory is specified the latter is default.

actual and **expected**
During the tests, models may be created and run on (possibly remote) CCM Core installations to create the expected and actual documents for comparison. The expected and actual part in the configuration section therefore serve to identify these CCM Cores. When the expected CCM Core is not specified, the server specified by the actual part is taken. The host may be specified either by the host name or its IP address. Optionally, you may indicate that a specific CCM Core is configured with a license for the CCM Platform:

```
<actual>
  <itpserver host="localhost"
             port="3001"
             ccm="true" />
</actual>
```

If you indicate that a specific CCM Core is configured for the CCM Platform, model execution runs on this CCM Core use a different script to test CCM document composition.

Test set descriptions

A test set description serves to combine several tests into one sequence typically handling one aspect of your regression test. A typical example would be all tests using the old doc format, all tests regarding letters or all tests pertaining to one particular database interaction. Such a test set is

described in a separate test set file which has the following typical form:

```
<?xml version="1.0" encoding="utf-8"?>
<hammer3testset name="letter_tests"
                dir="LetterTests"
                xmlns="http://www.aia-itp.com/Hammer3">
  <compose name="Begin-end end hash"
           type="Word"
           expecteddifferences="0">
           ...
  </compose>
  ...
</hammer3testset>
```

In the header of the test set, you specify the name of this particular test set and optionally a directory for the test sources. If this directory is not specified, the test directory from the Hammer3 test specification file is taken. If it is specified, it is taken as a relative path with regard to the directory given in your test specification file, unless you specify an absolute path.

The header of a test set is then followed by a number of compose sections, each describing a particular test.

A test set is referred to from the Hammer3 test specification file by providing the path to the test set description, usually taken relative to the test directory:

```
<testset src="Letters.hammer3" />
```

A compose section

A compose section serves to describe a single test to be run by Hammer3. Its heading provides the test name, the type of comparison (refer to Compare modes (see "Compare modes" page 10)) and the number of expected differences:

```
<compose name="VerifyWordCompareOneDiff"
         type="Word"
         expecteddifferences="1">
  <expected>
    <document src="expectedResultDocument.doc"/>
  </expected>
  <actual>
    <model src="testmodel.itp"
           keys="test key"
           extras="extra 1;extra 2" />
  </actual>
</compose>
```

In the above, the expected part and actual part describe the way that the expected and actual document must be constructed for the comparison. There are 5 ways of providing a document:

- as an ITP model;
- as an ITP model residing in a CCM Repository;
- as a template in an CCM Letterbook;
- as an ITP model document;
- or as an explicit document.

ITP model

The document may be described as an ITP model:

```
<model src="test.itp"/>
```

In this case Hammer3 will call the (expected or actual resp.) CCM Core to run the specified model. The result document of that model run will be the document used in the comparison.

Keys, extras, an CCM Core environment and data can be provided as well. These keys, extras and environment will be passed to the model. When these keys, extras or environment are not necessary, they may be omitted, in which case defaults are taken. The optional data attribute can be used to indicate a data file whose contents will be uploaded to fill the data backbone of the model during model execution.

```
<model src="test.itp"
       keys="c123456"
       extras="some extra"
       data="D:\temp\dbb.xml"
       environment="MyEnvironment"/>
```

Optionally one or more CCM Core configuration settings can be provided:

```
<model src="test2.itp">
  <configuration key="ITPLANGMODDOC"
                 value="NLD"/>
</model>
```

ITP model residing in an CCM Repository

As of version 4.2.0, CCM Core is able to retrieve models from an CCM Repository prior to running them. Hence it is possible to specify that the (expected or actual resp.) CCM Core should retrieve a model from a repository before CCM Core runs it to produce a result document:

```
<remotemodel src="/myproject/test" extras="extra 1;extra 2"
data="D:\temp\dbb.xml"/>
```

In this case Hammer3 will instruct the CCM Core to retrieve the model named "test" from the CCM Repository specified by the RepositoryServer setting in the default environment of the CCM Core, project "myproject". The object status is also taken from the RepositoryObjectStatus setting in the default environment. Instead of taking these parameters from an CCM Core environment, you may also specify a full repository path:

```
<remotemodel src="//test-o-mat2010:2586/myproject/test"
             extras="extra 1;extra 2"/>
```

If you specify a full repository path, the double slash before the host:port specification is required.

Similar to specifying a model, optional keys, extras, data and environment may be specified as well. For instance, if you wish to take your settings from another CCM Core environment, you specify the environment as attribute:

```
<remotemodel src="/myproject/test" environment="Test"/>
```

Template in an ITP Base Letterbook

CCM Core can run Logical Models from a CCM Content Publication Database described by a Template in an ITP Base Letterbook. It is possible to specify that CCM Core should run the template for the expected or actual model using a Letterbook URI:

```
<letterbookURI src="ltb:/mybook/test" extras="extra 1;extra 2"/>
```

In this case Hammer3 will instruct the CCM Core run the template named "test" in the Letterbook "mybook" registered with the ITP Base database configured for this CCM Core. Similar to remote models, additional parameters may be specified for the test run like extras or a specific environment, etc. The optional attribute "legacy" may be set to "true", to indicate that the provided data argument has a pre 4.2 data interface. It is also possible to specify a Logical Model directly:

```
<letterbookURI src="ltb:*MyLogicalModel" data="D:\temp\MyDbb.xml"/>
```

Version information
This feature is not available in version 4.2.1 or below.

ITP model document
When the document is given as an ITP model document, it must be specified as follows:

```
<modeldocument src="testModelDocument.doc"/>
```

With such a description, Hammer3 will call the (expected or actual resp.) CCM Core to make a model from the provided model document and subsequently run the freshly created model. Again the result document of that model run will be the document used in the comparison.

Similar to specifying a model, optional keys, extras and data may be provided as well as one or more CCM Core configuration settings:

```
<modeldocument src="testModelDocument.doc"
               keys="c123456"
               extras="some extra"
               data="D:\Temp\MyDBB.xml"
               environment="MyEnvironment">
  <configuration key="ITPLANGMODDOC"
                 value="NLD"/>
</modeldocument>
```

Explicit document
The expected as well as the actual document may also be given explicitly:

```
<expected>
  <document src="test (result).doc"/>
</expected>
```

In the above, references to the test sources (src="...."), are taken relative to the test directory, unless you provide an absolute path.

Compare modes

Hammer3 supports 4 compare modes:

- Word
- WordNoLayout

- ImagePerPixel
- ImagePerPage

Word compare mode

In the Word compare mode Hammer3 uses Microsoft Word to do the compare. Only changes that are found by Microsoft Word compare are reported. Microsoft Word is not always able to find 100% of the changes and not all changes result in a visually different document. For example, changes to style definitions are reported by Microsoft Word, even if the style is not used in the document. Changes to graphic elements such as using a different picture are not always found.

WordNoLayout compare mode

In the WordNoLayout compare mode Hammer3 uses Microsoft Word to do the compare. Sometimes when editing documents you introduce changes to styles that you as a user are not aware of and have no influence on the output. However in the Microsoft Word compare mode such changes show up as differences between the actual and the expected output. In certain manually created tests it is useful to ignore layout changes. For those circumstances the WordNoLayout compare mode is available. This is the most inaccurate compare mode. Only textual changes are caught, no layout changes or changes to graphic elements such as lines or artwork are found.

ImagePerPixel compare mode

As stated before, Microsoft Word does not always report differences between two documents, especially differences between embedded pictures are not reported. For this purpose, Hammer3 is also able to do an image compare between two result documents. By specifying that the type of comparison should be ImagePerPixel, CCM Core will use the Microsoft Word feature to convert the result of the (actual resp. expected) CCM Core run to XPS which is sent back the Hammer3 executable. Hammer3 will then convert these internally to TIFF and do an image compare between the two resulting in a difference document where every pixel is made red where there is a difference between the actual and expected document. The number of reported differences then is the number of pixels where there is a mismatch.

ImagePerPage compare mode

The ImagePerPage compare mode works in the same way as the ImagePerPixel compare mode, except that the number of reported differences is the number of pages where there is at least one pixel difference between the actual and expected document.

Hammer3 results

Hammer3 produces an xml file called results.xml in the configured results directory. This xml file has the following basic structure, namely a copy of the original Hammer3 configuration plus a list of test (compose) results:

```
<?xml version="1.0" encoding="utf-8"?>
<hammer3results xmlns="http://www.aia-itp.com/Hammer3">
  <configuration>
     ....
  </configuration>
  <setResult name="Combo.Base_tests"
             result="failure">
    <composeResult name="Base.test1"
                   type="Word"
                   result="success"
                   success="true" />
    </composeResult>
     ....
  </setResult>
  <setResult name="....">
     ....
  </setResult>
     ....
</hammer3results>
```

A set result embeds all results of the tests, belonging to a specific test set. A compose result gives the name of the particular test, the type of document comparison used for the test and the result of the test.

A test result may be one of the following four cases:

- 'success', denoting that the document comparison indeed gave the expected number of differences.
- 'failed', in which case the test was run, but the number of expected errors did not match the number of the actual errors of the document comparison.
- 'error' in which case there is something wrong with the specification of the test. The Hammer3 output during the run may then give some indications what was wrong with the specification.
- 'exception', meaning that the specific test encountered an exception. In this case, Hammer3 aborts gracefully with an error message, indicating the nature of the exception.

The other result output given, attribute success, is actually an condensed version of the result output. It will deliver the value 'true' for a successful run and 'false' otherwise. The result given in the set result header gives the worst result of all tests in that test set.

When the result of a test is 'failed', the expected document is copied to the results directory under the name 'testname'_expected, the actual document under the name 'testname'_actual and the difference document under the name 'testname'_compare to allow a further inspection of the actual differences. The extension of these results is taken from the produced document format.

Using Hammer3

If you want to specify tests you can create a Hammer3 file or a Hammer3 test set file using any text editor or xml file editor. The Hammer3 test specification file is an xml file. An xml schema file (hammer3.xsd) is available to validate these xml files.