DISCUSSION DRAFT
3-24-04

# OPEN XML COURT INTERFACE

Electronic Filing Manager
Development Iteration 1 – Test Report

March 24, 2004


# TABLE OF CONTENTS

# I. INTRODUCTION

# I. INTRODUCTION

This document is intended to provide a report on the outcome of the test phase associated with the first development iteration of the Open XML Court Interface (OXCI) Electronic Filing Manager (EFM) project. It will describe what was tested and report the results of those tests. The tests will be described in detail. A general overview of the test results will also be provided.

## A. RELATED DOCUMENTS

References to the following documents may be found within this design document.

| Title | Date | Source |
|---|---|---|
| OXCI EFM Design Document | January 2004 | counterclaim, Inc., MTG Management Consultants, L.L.C. |
| OXCI EFM Test Plan | March 2004 | counterclaim, Inc., MTG Management Consultants, L.L.C. |
| OXCI EFM JUnit HTML Report | March 2004 | counterclaim, Inc. |

## B. EXECUTIVE SUMMARY

One hundred eighty-six test cases are currently implemented in the OXCI EFM. Of these test cases, 174 execute successfully. Ten errors are reported and two failures are reported. Six of the incorrectly executing test cases are expected and six are unexpected. The unit tests currently verify that over 93 percent of the code is functioning correctly.

Over the next development iteration, all currently failing unit tests will be examined.

## II. TESTS CONDUCTED AND THEIR OUTCOMES

## II. TESTS CONDUCTED AND THEIR OUTCOMES

Each of the packages described in the OXCI EFM Test Plan are listed below. Typically, these packages represent modules defined in the OXCI EFM Design Document. The test cases associated with each package are described. The current status of the test case results is also discussed.

### A. ADMIN PACKAGE

The admin package contains objects related directly to the user interface. These objects are divided into three categories: action, form, and virtual objects. The objects generally conform with the Struts and Tiles-based Web application guidelines. A description of Struts and Tiles-based Web applications can be found in the OXCI EFM Design Document.

The action objects represent the application logic that is executed when some sort of action takes place in the user interface. An example of an action might be clicking on the "submit" button of a form in the application. The action associated with it will read the form data and make any core system component calls that it requires in order to decide what to do. It will then do what it determines is appropriate, and set the scene for the next user interface that will be displayed. Currently, eight different action objects contain the application logic for the EFM. The form and virtual objects are rather simple. They merely hold data to be used by the actions.

The form objects can also ensure that the data they contain meets specific conditions. Form objects are created when an HTML form element is used by the user interface.

The virtual objects are used by the actions to get data from previous actions and to store data for future actions to reference.

Since the package is rather complex, the test suite for it is also complex. The form and virtual objects are rather simple to test. Mainly the test cases associated with them simply check the data access functionality of the objects. The test cases for the actions are much more complex. They must construct a similar Web-like environment for the action to execute in.

Currently, there are 100 test cases evaluating the performance of the admin package. Thirty-three tests are associated specifically with actions. The remaining 67 tests are run on the form and virtual object classes. At this time, all tests execute successfully except for some tests associated with the action which manages filing events. Nine test cases associated with this action fail. Three of the failures are expected. These tests are expected to fail because the action accesses certain core EFM objects in a nontypical manner. These core EFM objects are not able to be set up in the test

environment in a manner which allows for their access. Six test cases are failing for unknown reasons. It is suspected that they are failing due to a bug in the testing code as opposed to a bug in the application logic. Currently, it is believed that certain test cases may be inserting incorrect data into the model thus causing other, seemingly unrelated, tests to fail. In certain instances, such as a fresh database, these six failing test cases will execute correctly. These failures will be addressed during the next development cycle.

In summary, 100 test cases function in this package and nine fail to complete successfully. Three of these failures are expected, and six of the failures are unexpected. All of the failures will be addressed during the second development iteration.

## B. BILLING PACKAGE

Three tests are provided for the billing package. The first test verifies the integrity of the billing response object. The tests ensure that data can be safely stored and retrieved from the object. The other two test cases deal with the specifics of the VeriSign billing module. The data storage methods are tested in one case and the processing functionality is verified in a separate test case.

All three test cases execute successfully.

## C. CLIENT PACKAGE

The three client objects implemented in this package are an HTTP filing client and SOAP and ebXML clients. The client objects are intended to be used by Electronic Filing Service Providers (EFSPs) which wish to easily file into the EFM. Five test cases ensure the proper function of these clients. The HTTP client test verifies that an HTTP submission can be successfully completed. The SOAP and ebXML objects are each more complicated then the HTTP filing client. The test cases associated with each of these clients ensure that data accessing methods all function correctly and that a filing submission can be successfully completed.

All five test cases execute successfully.

## D. CONFIG PACKAGE

The main objects of this package are the EFM rule set and the EFM configuration object. Two tests ensure the proper function of these implementations. One test ensures that the rule set can be

instantiated and passed on to the configuration object. The other test ensures that given the rule set, the configuration object executes correctly.

Both tests currently execute successfully.

## E. CONNECTOR PACKAGE

Two CmsConnector classes are provided which implement a Java Remote Method Invocation (RMI) connection. A test case is provided for each object. The remote object test case ensures that the data accessing methods behave properly. The local Java RMI test case attempts to make a connection to a remote object and submit a filing.

Currently only the test for the remote object executes successfully. The local object test results in an error due to the fact that an RMI environment is not set up for it at test time. The proper environment could be constructed, however the environment would change for each machine (IP address) on which the tests will be run. Thus, if it were to be modified to run successfully on counterclaim's test server, it may not run correctly on any other machine.

To summarize, one test executes successfully, and one encounters an error condition.

## F. CORE PACKAGE

The core package provides implementations of objects required by the EFM for core functionality. These objects represent entities such as the server and the individual components of the system. The base class for a component has a single test case which ensures the data accessing methods function correctly. The EFM controller object has two unit tests associated with it. One ensures the data accessing methods and the other tests the filing processing capabilities. The server object contains three tests. One test ensures the data accessors, another test verifies that the object can be instantiated correctly, and finally a test ensures the proper function of the server controls (such as starting and stopping).

Currently all tests except one reach successful outcomes. The test that reaches an error is the test that processes filings in the controller. The current test set up revolves around the RMI CmsConnector. Since the RMI is not configured in the test environment, this test encounters an error.

## G. COURT POLICY PACKAGE

This package provides implementations of the court policy object along with classes used to host and retrieve court policies. There are four test cases associated with this package. The simplest test case checks that the data stored in the configuration file regarding the court policy is successfully parsed and loaded in the court policy object. Two tests ensure that the court policy file can be hosted locally in the Web environment and can be read from a remote or local location. A fourth test ensures that the XPath queries executed on the court policy XML are correct.

All four of these tests currently execute correctly.

## H. LOGGER PACKAGE

The logger package provides a simple file system logger implementation. This module is used by the system when key events happen. A single test case verifies that method in the logger object executes without error.

Currently this test case executes successfully.

## I. VALIDATOR PACKAGE

The validator package handles the validation of XML instance documents. There are objects in the package that represent a validation problem and a handler for the problems. A single test case exists to ensure the data accessing methods of the validation problem. A single test case exists which verifies the proper execution of the methods contained within the problem handler object. Two test cases exist for both the Document Type Definition (DTD) validator and the XML Schema-based validator. One test ensures the data accessing methods of the validator classes while the other test ensures that they can successfully validate.

Currently six tests exist for this package and execute correctly.

## J. MODEL PACKAGE

The model package is large. It contains objects that represent data used by the EFM. These objects consist of items such as filings, documents, EFSPs, and courts. The package also contains managers for these objects that handle the creation, retrieval, and persistence of the objects.

Currently there are 43 separate test cases for this package. Many of the test cases ensure that the proper data accessing methods function correctly. A number of tests ensure that the manager objects successfully create and persist data objects. In general, application logic code is not stored in this package.

Forty-two tests currently execute successfully. One test fails. The failing test is actually a base test case which is extended by the other test cases in this package. This base test case is not intended to be run directly, however the testing suite that is currently implemented finds this test and tries to execute it.

## K. NOTIFICATION PACKAGE

The notification package provides a module capable of sending e-mail messages. One test case is provided with this package. The test case ensures that a valid e-mail message can be sent. Currently, the e-mail message is sent to *openefm@counterclaim.com*, and the test executes correctly.

## L. SECURITY PACKAGE

The security package contains simple classes for using a login name and password validation for EFSPs. A security violation object is also provided. Two test cases are available for this package. One test case ensures the data access methods of the security violation object function correctly. The other test case ensures that the functionality of the simple security module executes properly.

Currently, both test cases result in successful outcomes.

## M. TRANSCEIVER PACKAGE

The transceiver package contains classes that accept a court filing and pass the filing on to the core EFM components. A new transceiver is implemented for each protocol used to pass a filing to the EFM. Currently, three protocols are supported. These are HTTP, SOAP, and ebXML. Test cases exist for each implementation. The test cases ensure that any data accessing methods function correctly. The test cases also ensure that the transceiver objects receive filings and that these filings are properly passed to the core EFM components.

There are currently seven unit tests associated with this package. At present, they all execute successfully.

## N. UTIL PACKAGE

The util package contains helper functions for the rest of the application to use. The methods are executed independently from any system-specific task. They are used by other code modules and are isolated from the modules that use them since many different modules may include references to these objects. Currently, they mainly provide input and output routines associated with file access.

Three test cases are provided for the util package. These test cases ensure the proper execution of the individual IO routines. At this time, these tests all execute correctly.

# APPENDIX A
# REVISION HISTORY

## REVISION HISTORY

| Version | Date | Revised By | Description |
|---|---|---|---|
| 0.1 | 3-24-04 | Mr. Jim Beard counterclaim, Inc. | Initial creation of the document as an outline. |