

**OASIS Committee Note**

Implementation Guidance for Electronic Court Filing Version 4.1

Committee Note 01

17 August 2023

This stage:

<https://docs.oasis-open.org/legalxml-courtfiling/ecf-guide/v4.1/cn01/ecf-guide-v4.1-cn01.docx> (Authoritative)

<https://docs.oasis-open.org/legalxml-courtfiling/ecf-guide/v4.1/cn01/ecf-guide-v4.1-cn01.html>

<https://docs.oasis-open.org/legalxml-courtfiling/ecf-guide/v4.1/cn01/ecf-guide-v4.1-cn01.pdf>

Previous stage:

N/A

Latest stage:

<https://docs.oasis-open.org/legalxml-courtfiling/ecf-guide/v4.1/ecf-guide-v4.1.docx> (Authoritative)

<https://docs.oasis-open.org/legalxml-courtfiling/ecf-guide/v4.1/ecf-guide-v4.1.html>

<https://docs.oasis-open.org/legalxml-courtfiling/ecf-guide/v4.1/ecf-guide-v4.1.pdf>

Technical Committee:

[OASIS LegalXML Electronic Court Filing TC](https://www.oasis-open.org/committees/legalxml-courtfiling/)

Chair:

James Cabral (jim.cabral@infotrack.com), [InfoTrack US](https://www.infotrack.com/)

Editors:

James Cabral (jim.cabral@infotrack.com), [InfoTrack US](https://www.infotrack.com/)

Gary Graham (GGraham@courts.az.gov), [Arizona Supreme Court](http://www.azcourts.gov/)

Related work:

This document is related to:

* *Electronic Court Filing Version 4.1.* Edited by James Cabral, Gary Graham, and Philip Baughman. Latest stage: <https://docs.oasis-open.org/legalxml-courtfiling/ecf/v4.1/ecf-v4.1.html>.

Abstract:

This committee note provides non-normative guidance to implementers of the LegalXML Electronic Court Filing Version 4.1 specification.

Status:

This is a Non-Standards Track Work Product. The patent provisions of the OASIS IPR Policy do not apply.

This document was last revised or approved by the OASIS LegalXML Electronic Court Filing TC on the above date. The level of approval is also listed above. Check the "Latest stage" location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at <https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=legalxml-courtfiling#technical>.

TC members should send comments on this document to the TC’s email list. Others should send comments to the TC’s public comment list, after subscribing to it by following the instructions at the “[Send A Comment](https://www.oasis-open.org/committees/comments/index.php?wg_abbrev=legalxml-courtfiling)” button on the TC’s web page at <https://www.oasis-open.org/committees/legalxml-courtfiling/>.

Citation format:

When referencing this document, the following citation format should be used:

[ECF-Guide-v4.1]

*Implementation Guidance for Electronic Court Filing Version 4.1*. Edited by James Cabral and Gary Graham. 20 June 2023. OASIS Committee Note 01. <https://docs.oasis-open.org/legalxml-courtfiling/ecf-guide/v4.1/cn01/ecf-guide-v4.1-cn01.html>. Latest stage: <https://docs.oasis-open.org/legalxml-courtfiling/ecf-guide/v4.1/ecf-guide-v4.1.html>.

Notices:

Copyright © OASIS Open 2023. All Rights Reserved.

Distributed under the terms of the OASIS IPR Policy, [<https://www.oasis-open.org/policies-guidelines/ipr/>]. For complete copyright information please see the Notices section in an Appendix below.

Table of Contents

[1 Introduction 5](#_Toc143170561)

[1.1 Changes from earlier Versions 5](#_Toc143170562)

[1.2 Glossary 5](#_Toc143170563)

[1.2.1 Definitions of terms 5](#_Toc143170564)

[1.2.2 Acronyms and abbreviations 6](#_Toc143170565)

[1.2.3 Document conventions 6](#_Toc143170566)

[2 Installation 7](#_Toc143170567)

[2.1 Environment 7](#_Toc143170568)

[2.1.1 Relative Paths and schemaLocation 7](#_Toc143170569)

[2.1.2 Working Environment Set-up 7](#_Toc143170570)

[3 Backward Compatibility 8](#_Toc143170571)

[3.1 Version Differences 8](#_Toc143170572)

[3.1.1 Portable Media SIP Deprecated 8](#_Toc143170573)

[3.1.2 New Web Services SIP 8](#_Toc143170574)

[3.1.3 Bulk and Batch Filing Terminology 9](#_Toc143170575)

[3.1.4 Court Policy Changes 9](#_Toc143170576)

[3.1.5 New Wrappers Schema 10](#_Toc143170577)

[3.1.6 Normative Operations Signatures 11](#_Toc143170578)

[3.1.6.1 Relaxed Operation Signature Cardinality 11](#_Toc143170579)

[3.1.7 Optional Callback Messages 11](#_Toc143170580)

[3.1.8 Required and Optional Operations 12](#_Toc143170581)

[3.1.8.1 NotifyDocketingComplete 12](#_Toc143170582)

[3.1.8.2 NotifyFilingReviewComplete 12](#_Toc143170583)

[3.1.8.2.1 Providing Clerk Review and Other Results 12](#_Toc143170584)

[3.1.8.3 RecordFiling 14](#_Toc143170585)

[3.1.9 GetFeesCalculation 14](#_Toc143170586)

[3.1.10 NIEM-Core Schema Changes 15](#_Toc143170587)

[3.1.11 New Country Codes Code List 15](#_Toc143170588)

[3.2 Backward Compatibility Issues and Considerations 15](#_Toc143170589)

[3.2.1 Normative Operation Signatures 16](#_Toc143170590)

[3.2.2 Parameter Cardinalities 16](#_Toc143170591)

[3.2.3 Backward Compatibility Considerations 16](#_Toc143170592)

[3.2.4 Backward Compatibility Summary 18](#_Toc143170593)

[4 Limitations, Caveats and Other Items of Note 19](#_Toc143170594)

[4.1 Use of “Filing” Terminology 19](#_Toc143170595)

[4.2 Filing Identifier 20](#_Toc143170596)

[4.3 Multiple CoreFilingMessages in ReviewFilingRequest 20](#_Toc143170597)

[4.3.1.1 Multi-Episode Clerk Review with multiple CoreFilingMessages 22](#_Toc143170598)

[4.3.1.2 RecordFilingRequests for multiple CoreFilingMessages 22](#_Toc143170599)

[4.4 Service List Registry 23](#_Toc143170600)

[4.5 Hub Service 23](#_Toc143170601)

[4.6 Implementation Namespace 23](#_Toc143170602)

[4.7 Payment Maximum Amount 24](#_Toc143170603)

[4.8 ECF 5 Alignment 24](#_Toc143170604)

[4.9 NIEM Version 24](#_Toc143170605)

[4.10 ECF Conformance and Compliance 25](#_Toc143170606)

[Appendix A. Informative References 28](#_Toc143170607)

[Appendix B. Acknowledgments 29](#_Toc143170608)

[B.1 Special Thanks 29](#_Toc143170609)

[B.2 Participants 29](#_Toc143170610)

[Appendix C. Wrappers.xsd structures 31](#_Toc143170611)

[Appendix D. Adapting ECF v4.1 Web Services SIP to ECF 4.01 33](#_Toc143170612)

[D.1 Summary 33](#_Toc143170613)

[D.2 Introduction 33](#_Toc143170614)

[D.3 Rationale 33](#_Toc143170615)

[D.4 Considerations 35](#_Toc143170616)

[D.5 Artifact Modifications 36](#_Toc143170617)

[D.6 Examples 38](#_Toc143170618)

[D.6.1 Example 1 - ECF 4.01 adapted wrappers.xsd 38](#_Toc143170619)

[D.6.2 Example 2 - ECF 4.01 adapted CourtRecordMDE.wsdl 43](#_Toc143170620)

[D.6.3 Example 3 - ECF 4.01 adapted FilingAssemblyMDE.wsdl 45](#_Toc143170621)

[D.6.4 Example 3 - ECF 4.01 adapted FilingReviewMDE.wsdl 47](#_Toc143170622)

[D.6.5 Example 3 - ECF 4.01 adapted ServiceMDE.wsdl 50](#_Toc143170623)

[D.6.6 Example 3 - ECF 4.01 adapted CourtRecordMDE-ImplementationExample.wsdl 52](#_Toc143170624)

[D.6.7 Example 3 - ECF 4.01 adapted FilingAssemblyMDE-ImplementationExample.wsdl 52](#_Toc143170625)

[D.6.8 Example 3 - ECF 4.01 adapted FilingReviewMDE-ImplementationExample.wsdl 53](#_Toc143170626)

[D.6.9 Example 3 - ECF 4.01 adapted ServiceMDE-ImplementationExample.wsdl 53](#_Toc143170627)

[Appendix E. Revision History 55](#_Toc143170628)

[Appendix F. Core Specification Normative Requirements 56](#_Toc143170629)

[Appendix G. Example Title 60](#_Toc143170630)

[G.1 Subsidiary section 60](#_Toc143170631)

[G.1.1 Sub-subsidiary section 60](#_Toc143170632)

[G.1.1.1 Sub-sub-subsidiary section 60](#_Toc143170633)

[G.1.1.1.1 Sub-sub-sub-subsidiary section 60](#_Toc143170634)

[Appendix H. Notices 61](#_Toc143170635)

# Introduction

This document is a Technical Committee Note and is not a Specification document.

This document addresses Committee Specifications for Electronic Court Filing v4.1 and Web Services Service Interaction Profile v4.1.

This document provides implementation guidance for ECF v4.1 and Web Services SIP v4.1.

Since the Technical Committee (TC) anticipates that implementors of the v4.1 specifications will have previously implemented prior ECF versions, especially ECF v4.x and its supporting Web Services SIP. As such, this document will identify differences between the new v4.1 specifications and its immediate prior ECF specifications.

This document will also address other implementation issues.

The TC may use this Committee Note to provide specification clarifications and to make non-normative suggestions or recommendations to implementors.

Since this document is not a specification, any inadvertent discrepancies or contradictions are governed by the normative requirements of the actual specification and specification documents.

## Changes from earlier Versions

This section provides a description of significant differences from previously published, differently numbered Versions of this specification, if any. (Detailed revision history of this numbered Version should be tracked in an Appendix.)

This is an initial version. No prior published versions of this document exist.

## Glossary

### Definitions of terms

This section defines key terms used in this Committee Note:

**ECF Message**

An XML instance of one of the XML structures defined by schema provided in the ECF provided xsd/message folder. ECF provides 24 message structures. Although ECF Messages are specified as MDE operation parameters, ECF Messages are not provided directly to operations. The ECF Messages are wrapped within request and response structures for exchanges.

**Exchange**

A communication between MDEs initiated by means of an XML instance provided as an invocation parameter to an MDE operation and resulting in a synchronous XML response.

**Submission**

An individual e-filing tendering provided by the FAMDE to the FRMDE as a single transaction (e.g., by a single invocation of the ReviewFiling operation); i.e., a ReviewFilingRequest.

### Acronyms and abbreviations

|  |  |
| --- | --- |
|  |  |
| **CRMDE** | Court Record Major Design Element |
| **ECF** | Electronic Court Filing – this acronym generally refers to the OASIS LegalMXL Electronic Court Filing Technical Committee or one of this committee’s specifications. |
| **FAMDE** | Filing Assembly Major Design Element |
| **FRMDE** | Filing Review Major Design Element |
| **MDE** | Major Design Element |
| **MRM** | MessageReceiptMessage |
| **NIEM** | National Information Exchange Model  |
| **RvFR** | ReviewFilingRequest |
| **SIP** | Service Interaction Profile |
| **TC** | Technical Committee - this acronym refers to the OASIS LegalMXL Electronic Court Filing Technical Committee |
| **URI** | Uniform Resource Identifier |

### Document conventions

* Naming conventions
* Font colors and styles
* Typographical conventions
* XML element and attribute names are displayed in Courier New font, e.g., nc:DocumentIdentification
* Literal quotations are contained with double-quote characters and are highlighted using blue text.

# Installation

This section provides suggestions for the installation of ECF v4.1 and ECF Web Services SIP v4.1.

## Environment

A working ECF v4.1 environment permits review of the specification and technical artifacts. This section provides instructions for setting up a working ECF v4.1 environment.

Implementors may choose to alter the locations of files and/or use alternative folder structures and names in an actual production environment. Before making this choice, it should be noted that the various ECF artifacts are deployed using relative folder path references. Revisions to ECF provided artifacts may be necessary if the production environment is different that the ECV v4.1 working environment.

### Relative Paths and schemaLocation

ECF uses relative path designators in artifacts provided with specifications, typically for import and schemaLocation.

In doing so, all artifacts provided with the Core specification within the deployment zip file are properly located, one to another, within the zip file. This makes setting up a working ECF v4.1 environment, based on the Core specification, relatively easy.

However, when including Profiles, such as the Web Service SIP or Signature Profile, then additional steps may be required.

### Working Environment Set-up

By following the following instructions provided, ECF artifacts will be placed into proper folder locations so that the relative paths provided are correctly positioned. As such, artifacts will properly reference one another (e.g., in schemaLocation, etc.).

To set-up a working ECF v4.1 environment:

a. Determine a root folder location

b. Unzip (e.g., Extract All) from the Core specification zip file into your root folder

b1. When unzipping, remove the rightmost folder name from the suggested path

c. Unzip (e.g., extract) the corresponding Web Services SIP files into the root folder

c.1 Once unzipped, then cut the wsdl folder and paste it beneath the Core specification root folder at the same folder level as the xsd folder.

# Backward Compatibility

This section addresses the differences between this version of ECF and the prior version with the same major version number, and backward compatibility implications arising from these differences.

The ECF v4.1 Core specification states:

“This specification does not assume that prior specifications will be deprecated.  However, ECF 4.1 is not guaranteed to be backward-compatible with previous versions including ECF 4.0 and 4.01, both based on NIEM 2.x.  Applications based on ECF versions which themselves are based on NIEM versions other than NIEM 2.x (such as ECF 3.0, 3.01 and 3.1 specifications) will certainly not interoperate successfully with applications using this specification.  This fact is indicated by the assignment of a new major and minor version number to the specifications.”

OASIS describes backward compatibility as:

• “Backward compatibility: A standard is said to allow backward compatibility, if products designed for the new standard can receive, read, view or process older standards or formats. Or, it is able to fully take the place of an older product, by inter-operating with products that were designed for the older product.”

This section addresses ECF v4.1 backward compatibility to ECF v4.01 Errata 2.

## Version Differences

This section identifies notable differences between ECF v4.01 Errata 2 and ECF v4.1.

### Portable Media SIP Deprecated

ECF 4.1 (i.e. [ECF-WS-SIP-v4.1]) deprecates use of the Portable Media SIP.

### New Web Services SIP

ECF 4.1 introduces a new ECF 4.1 Web Services SIP. Prior Web Services SIP versions are not supported by ECF v4.1.

The prior Web Services SIP (v2.01) (i.e., [ECF-v4.0-WS-SIP-v2.01]) provides a single WSDL for use for all MDEs and all MDE operations. The new Web Services SIP v4.1 provides separate WSDL for each MDE.

WS-SIP v2.01 declares a single XML namespace:

• urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:WebServicesProfile-2.0

In WS-SIP v2.01 the above namespace URI is also used as the Service Interaction Profile Identifier as specified in Section 2.1 ‘Service Interaction Profile Identifier’.

WS-SIP v4.1 declares four (4) namespaces:

• urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.1

• urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.1

• urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingReviewMDE-4.1

• urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:ServiceMDE-4.1

In WS-SIP v4.1, a single separate URI is used as the Service Interaction Profile Identifier:

• urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:WebServices-4.1

WS-SIP v4.1 incorporates the new Core specification wrappers.xsd schema from the Core ECF v4.1 specification.

### Bulk and Batch Filing Terminology

WS-SIP v4.1 no longer claims to support ‘bulk filings’. The ECF TC has elected to discontinue use of the terminology of “Bulk filings” and “Batch filings” as, at this time, there does not appear to be industry/legal community-wide consensus on the definition of these terms.

### Court Policy Changes

Court Policy – Very little has been changed in Court Policy. These minor changes are:

a. The specification makes it clear that some form of machine-readable court policy is required in a complete implementation. However, it is not necessary that machine-readable court policy is implemented in an ECF suggested manner, e.g., by providing the GetPolicy operation that utilizes the ECF-4.1-CourtPolicyQueryMessage and CourtPolicyResponseMessage.

When a non-ECF machine-readable court policy is implemented, the Core specification provides broad implementor discretion. However, implementors should be mindful of code-lists detailed in Section 2.4.4 ‘Court-specific Code Lists’.

b. [ECF-WS-SIP-v4.1] specification, Section 2.1 was corrected from the prior version specification to specify the CourtPolicyResponseMessage and not the errantly specified CourtFilingResponseMessage.

c. The new element <RequireAsynchronousResponsesIndicator> has been added to the CourtPolicyResponseMessage and to machine-readable court policy.

When this element is ‘false’ then asynchronous callback messages need not be provided for any NotifyDocketingComplete operation or for any NotifyFilingReviewComplete operation. Take note that the Core specification does not require suppression of both the RecordDocketingCallbackMessage elements and ReviewFilingCallbackMessage/PaymentReceiptMessage elements when <RequireAsynchronousResponsesIndicator> is ‘false’. An implementation may choose to suppress RecordDocketingCallbackMessage elements while continuing to provide ReviewFilingCallbackMessage/PaymentReceiptMessage.

Since ECF machine-readable Court Policy is optional (although some form of machine-readable court policy is mandatory) then the <RequireAsynchronousResponsesIndicator> element may not be relevant. When the <RequireAsynchronousResponsesIndicator> element is not used, then this element should not be understood to be ‘false’ nor should it be presumed to be ‘true’. In this circumstance, it is recommended that the presumed value should be stated in Human-readable court policy.

When <RequireAsynchronousResponsesIndicator> is ‘true’, then <SendingMDELocationID> and <SendingMDEProfileCode> MUST be included (with appropriate values) in all messages that provide these elements, such as CoreFilingMessage, CaseListQueryMessage, CourtPolicyQueryMessage, FeesCalculationQueryMessage, RecordDocketingCallbackMessage, etc.

### New Wrappers Schema

Wrappers.xsd is new with ECF 4.1.

Although use of wrappers.xsd is not normative in ECF v4.1, its use is required when also using [ECF-WS-SIP-v4.1].

Wrappers.xsd introduces request and response structures. These request and response structures fill a gap that has existed between the Core specification and the Web Services SIP specifications in prior ECF versions. This gap, by necessity, was typically filled in implementations by defining and using implementation specific exchange schema or by WSDL modifications or extensions.

One consequence is that terminology, such as ‘ReviewFilingRequest’, now has a more specific technical meaning in ECF v4.1. (i.e., ReviewFilingRequest is a complex element that contains one-to-many CoreFilingMessage elements and zero-to-one PaymentMessage element as defined in wrappers.xsd).

However, even though ‘request’ and ‘response’ structures are defined in wrappers.xsd, as ‘types’ (e.g., ‘GetPolicyRequestType’), there may not be corresponding ‘Request’ named elements for these types. However, all elements derived from response Type Structures are named such that the element name ends with ‘Response’.

For example, although there is a “GetPolicyRequestType” defined in wrappers.xsd, there is not a ‘GetPolicyRequest’ element defined in wrappers.xsd or in any other schema or WSDL. Instead, the GetPolicy element is derived from GetPolicyRequestType. However, wrappers.xsd defines both a ReviewFilingRequestType element and a ReviewFilingRequest element derived from corresponding type structures.

Also, one request type structure was named without including ‘Request’ within the type-name (i.e., ‘GetDocumentType’ would be better named as ‘GetDocumentRequestType’). The corresponding operation response type structure (i.e., GetDocumentResponseType) is the basis for the response element DocumentResponseMessage.

See Appendix C for a list of all request and response type structures defined in wrappers.xsd.

### Normative Operations Signatures

In prior ECF Core specifications, MDE operation signatures were not normatively defined, however ‘suggested’ (e.g., informative) operations signatures were provided in an Appendix (as Appendix C in prior specification documents). Now in ECF 4.1, operation signatures, provided in Section 5 ‘MDE Operations’, are normative. These operation signatures define the ECF message parameters, the cardinality for the parameters, and the order in which the parameter messages must be provided. The response output message is also specified.

#### Relaxed Operation Signature Cardinality

Operation signature parameter cardinalities have been relaxed for some operations. Specifically:

1. ReviewFiling now permits multiple CoreFilingMessage elements.
2. RecordFiling now permits multiple CoreFilingMessage elements.
3. NotifyDocketingComplete now permits multiple RecordDocketingCallbackMessage elements. Additionally, RecordDocketingCallbackMessage may now contain multiple ReviewedLeadDocument elements.
4. NotifyFilingReviewComplete now permits multiple ReviewFilingCallbackMessage elements. Additionally, ReviewFilingCallbackMessage may now contain multiple ReviewedLeadDocument elements.
5. FeesCalculationQueryMessage now permits multiple CoreFilingMessage elements.

Although operation signatures are defined using ECF message structures as parameters, these normative parameters are bound within a request structure defined within wrappers.xsd.

New, future ECF v4.1 compatible Service Interaction Profiles may or may not employ the request and response structures defined in wrappers.xsd. For example, if/when the IBM MQ Service Interaction Profile is updated, this specification may or may not incorporate wrappers.xsd. However, the operation signatures and cardinalities provided in section 5 ‘MDE Operations’ must be adhered to.

### Optional Callback Messages

Asynchronous callback messages are now optional for:

• NotifyDocketingComplete

• NotifyFilingReviewComplete

The ECF TC observed that some courts preference was for ‘auto-clerk-review’ acceptance of e-filing submissions for some or all matter types. ‘Auto-Accepted’ submissions are not manually clerk reviewed, and are directly docketed into the CRMDE, typically from the ReviewFilingRequest and not through a RecordFilingRequest.

### Required and Optional Operations

The NotifyDocketingComplete operation and the NotifyFilingReviewComplete operation are no longer required operations. Also, depending upon the implementation, RecordFiling may also not be required.

However, when the <RequireAsynchronousResponsesIndicator> element in machine-readable Court Policy is ‘true’ and when the implementation is using the RecordFiling operation (see 3.1.8.3) then the NotifyDocketingComplete operation must be invoked.

Additionally, when the <RequireAsynchronousResponsesIndicator> element in machine-readable Court Policy is ‘true’ then the NotifyFilingReviewComplete operation must be invoked.

#### NotifyDocketingComplete

The NotifyDocketingComplete operation is now optional in ECF v4.1

With ECF v4.1, implementations that do not employ the RecordFiling operation (e.g., in an ECF conformant manner) also need not invoke the NotifyDocketingComplete operation. Additionally, implementations in which <RequireAsynchronousResponsesIndicator> is ‘false’ also need not invoke NotifyDocketingComplete.

For clarity, it should be noted that the NotifyDocketingComplete operation is a required operation in prior ECF versions, even when a submission is fully rejected in Clerk Review (see ECF 4.01 Section 3.2.7 ‘NotifyDocketingComplete’ which states: “The Court Record MDE MUST invoke the NotifyDocketingComplete operation on the Filing Review MDE as a callback message to the RecordFiling operation to indicate whether the filing was accepted or rejected by the court record system. If the Court Record MDE rejected the filing, an explanation MUST be provided)”.

How filing acceptance or rejection, and other docketing information is communicated to the Filing Review MDE is unspecified. The Core specification is silent on this. But it seems clear that if the clerk review results are to be made available to the FRMDE, then this information must be provided by means other than through the NotifyDocketingComplete operation, when the NotifyDocketingComplete operation is not used.

#### NotifyFilingReviewComplete

As stated previously, the NotifyFilingReviewComplete operation is now optional.

When the NotifyFilingReviewComplete operation is not utilized, then providing clerk review results, payment receipt information, and docketing information is challenging. This is considered next:

##### Providing Clerk Review and Other Results

NotifyFilingReviewComplete provides clerk review results, docketing results, and payment receipt information to the FAMDE. So, when NotifyFilingReviewComplete is not utilized then how does this information get communicated back to the FAMDE?

The Core specification is silent on this. But it seems clear that if the clerk review results are to be made available to the FAMDE, then this information must be provided by means other than through the NotifyFilingReviewComplete operation, when the NotifyFilingReviewComplete operation is not used.

It may be notable to observe that, although implied, the Core specification is not clear whether it is a requirement that the clerk review results for a submission must be communicated to the FAMDE, but this is a recommended practice. Section 2.2 ‘Major Design Elements’, bullet 2 ‘Filing Review MDE’ states that FilingReview “enables a court to receive and review a filing message and prepare the contents for recording in its case management and document management systems, sending a response concerning the filing to the Filing Assembly MDE”. The words “sending a response” at a minimum implies or suggests that the NotifyFilingReviewCompleteRequest should be sent.

One option for providing clerk review results to the FAMDE, when not providing NotifyFilingReviewCompleteRequest, is through the use of the GetFilingStatus operation. GetFilingStatus can return a single FilingStatusCode (e.g., ‘received’, ‘accepted’, ‘partially-accepted’, or ‘rejected’). GetFilingStatus can also return limited docketing information in nc:CaseDocketID and nc:Case. Additional case and docketing information may be obtained using the GetCase operation. There are no ECF message queries that will provide payment receipt information.

If the GetFilingStatus request specifies a single ‘filing’ (e.g., by providing a ‘filing-identifier’ as a query parameter, e.g., by using nc:DocumentIdentification) then a FilingStatusCode can be returned for this ‘filing’.

If, however, the filing status query parameter identifies something other than a single ‘filing’ (e.g., the query parameter provides a ‘Case ID’ instead of a filing-identifier) then multiple ‘filings’ may qualify, yet only one ecf:FilingStatus can be returned in the response, and only a single response can be returned for a request.

There is apparent flexibility regarding query parameters permitted for GetFilingStatus. Section 3.2.10 ‘GetFilingStatus’ says “the Filing Assembly MDE MAY invoke the GetFilingStatus query operation with the filing Identifier”. Although this specification statement may not be normative, it at least suggests that ‘filing-identifier’ is a preferred or recommended query parameter.

However, the element documentation for statusquery:FilingStatusQueryMessage says “this is query to get a filing’s status by Filer Identification, CaseID, or Filing Number”. ‘Filing Number’ is understood to mean the ‘filing identifier’. This element documentation suggests that query parameters, other than filing-identifier, would not be disallowed by specification. When using parameters other than filing-identifier, multiple ‘filings’ may result.

Providing clerk review results can be further complicated when multi-episode clerk review is allowed. For example, when a submission contains a single CoreFilingMessage which in turn contains multiple FilingLeadDocument elements, and clerk review has been concluded for some, but not all lead documents, then at that moment, what is the filing status for the submission? The answer, choosing one of the four ECF provided code options, would presumably be “partially-accepted”. However, if instead of accepting the documents in the first multi-episode clerk review session, the reviewed documents were rejected, then what is the filing status? ECF does not provide an “partially-rejected” filing status. If each result (e.g., RFR) for a concluded multie-epsisode clerk review episode is considered a ‘filing’, then the filing status in this rejection circumstance may be “rejected’ and not “partially-rejected”.

Now that ECF 4.1 allows multiple CoreFilingMessage(s) within a single ReviewFilingRequest, the option of multi-episode clerk review can take on a whole new dimension. See section 3.3.2.1 ‘Multi-Episode Clerk Review with multiple CoreFilingMessages’ later in this document for additional information on this topic.

Of course, clerk review results, payment receipt information, and docketing information cannot be provided to the FAMDE from the FRMDE unless the FRMDE has access to this information. Prior to ECF v4.1, the NotifyDocketingComplete operation was used to provide most of this information to the FRMDE (note: NotifyDocketingComplete does not provide payment receipt information to the FRMDE). Now that NotifyDocketingComplete is optional in ECF 4.1, implementations that do not support this operation must provide some other method or methods to inform the FRMDE. These methods appear to be outside the ECF v4.1 specification.

The method for providing payment receipt information has never been fully addressed within ECF specifications. The changes in ECF v4.1 do not impact this understanding.

The PaymentMessage is provided to the FRMDE in the RvFR. Payment and payment receipt information is not provided to the CRMDE in the RFR. Additionally, payment information and payment receipt information are not included in the NDC. Since the NotifyFilingReviewCompleteRequest provides PaymentReceiptMessage and much or all the information provided in PaymentReceiptMessage originated in the PaymentMessage it must be presumed that a stateful protocol is envisioned.

Implementations that prefer a stateless protocol may consider extending the RFR and NDC exchanges to include payment and/or payment receipt information.

#### RecordFiling

ECF defines a RecordFiling operation on the CRMDE. As specified, the RecordFiling operation consumes one to multiple RecordDocketingMessage elements and one to multiple CoreFilingMessage elements as input parameters, returning a synchronous MessageReceiptMessage.

Although the Core specification illustrates the RecordFiling operation as required, through the use of bold characters in Section 3.1 ‘The Filing-Preparation-to-Docketing Process Model’ and by not including RecordFiling within the shaded ‘opt.’ (i.e., optional) rectangle in Figure 4 within the same section, this section also provides the following statement:

“when the RecordFiling operation has been implemented within the same system as the ReviewFiling operation, then the RecordFiling operation need not be provided in an ECF 4.1 compliant manner.”

ECF provides this flexibility to accommodate implementations in which the functions attributed to the RecordFiling operation are fulfilled by capabilities inherent to the CRMDE system. Typically, this is a Case Management System that provides e-filing clerk review capabilities. Although the ECF Core “specification is not intended to define how operations must be implemented”, the RecordFiling operation is understood to provide case ‘docketing’ functions. The details of ‘docketing’ are court and implementation specific.

### GetFeesCalculation

ECF 4.1 has been modified to allow more than one CoreFilingMessage within a GetFeesCalculation request. This has been done to support newly expanded ReviewFilingRequest elements that may now also provide more than one CoreFilingMessage.

When a ReviewFilingRequest provides multiple CoreFilingMessage elements, then a single GetFeesCalculation request can provide all of these CoreFilingMessage elements within the single request and get back a single (possibly aggregate) FeesCalculationAmount, in the FeesCalculationResponseMessage, for the collection of provided CoreFilingMessage elements.

Alternatively, multiple GetFeesCalculation requests can be submitted (e.g., one for each CoreFilingMessage) resulting in separate FeesCalculationAmount elements. Since a ReviewFilingRequest still only provides a single PaymentMessage, then these separate FeesCalculationAmount elements may need to be summed to provide a single PaymentMessage AllowanceCharge Amount or may be listed individually as separate AllowanceCharge elements.

Although section 3.2.3 GetFeesCalculation specifies that “The Filing Assembly MDE MAY query for the fees associated with a filing by invoking the MDE’s GetFeesCalculation operation, with a filing as a parameter” (i.e., singular) this should be understood as requiring at least one CoreFilingMessage, but also permitting multiple CoreFilingMessage elements as invocation parameters.

Note that in the new Section 5 ‘MDE Operations’, the Parameters listed in Section 5.2.1 shows ‘FeesCalculationQueryMessage’ as the input parameter for the GetFeesCalculation operation (and not CoreFilingMessage). FeesCalculationQueryMessage requires at least one CoreFilingMessage but allows multiple CoreFilingMessage elements.

### NIEM-Core Schema Changes

NIEM-Core.xsd changes

• nc:ItemOtherIdentification within nc:ItemType – maxOccurs changed from “1” to “unbounded”.

• nc:ObligationEntity within nc:ObligationType - maxOccurs changed from “1” to “unbounded”.

• nc:OrganizationIdentification - maxOccurs changed from “1” to “unbounded”.

• LocationCountryISO3166Alpha2Code added – can be substituted for nc:LocationCountry.

• PersonCitizenshipISO3166Alpha2Code added – can be substituted for nc:PersonCitizenship.

### New Country Codes Code List

The schema iso\_3166.xsd has been added.

This schema provides two-letter country codes.

## Backward Compatibility Issues and Considerations

This section addresses issues of backward compatibility with ECF v4.01.

### Normative Operation Signatures

Operation signatures are now normatively defined.

ECF 4.01 implementations that support ECF operations with input and output parameters that are not conformant with the signatures defined in Section 5 will require modifications to those operations for ECF v4.1 compatibility.

If ECF v4.01 implementations follow the informative guidelines provided in Appendix C of the ECF v4.01 Core specification, then these prior informative operation signatures have been replaced by normative operation signatures that are largely consistent with those of the prior ECF version.

Some ECF v4.1 operation signatures have been modified to relax cardinality restrictions and allow multiple ECF message parameters.

### Parameter Cardinalities

The change in ECF operation invocation parameter cardinality revisions as identified in the prior section above, may introduce backward compatibility issues.

Since these cardinality changes are optional, implementations that do not take advantage of these newly allowed multiplicities are not expected to experience any issues.

However, existing implementations that have exercised implementation/court specific extension options, may require modifications. These local extensions should be reviewed.

If the court/implementation extension was made to provide the same relaxed parameter cardinalities as provided in ECF v4.1, then the court/implementation could retire that extension.

### Backward Compatibility Considerations

Could an ECF v4.01 MDE (e.g., FAMDE) successfully provide an ECF v4.01 ReviewFilingRequest RvFR) to an ECF v4.1 MDE (e.g., FRMDE)?

This question must be considered in two parts, a) for the Core ECF specification and b) for the Web Services WSDL.

a. If an ECF v4.1 FRMDE received an ECF v4.01 RvFR, could it be successfully processed, and could the ECF v4.01 FAMDE successfully process the synchronous ECF v4.1 ReviewFilingResponse?

Of course, to truly answer this question, testing, for each specific implementation circumstance, would be required.

But generally, changes (especially XML changes) are considered to be backward compatible if they relax specifications and do not tighten specifications. For example, if an element that had been previously required is made optional, then this is a relaxation and is considered to be backward compatible. Doing the opposite would not be backward compatible.

By and large, the ECF v4.1 XML schema modifications have relaxed criteria and have not tightened constraints. These include:

• Adding optional wrappers.xsd

• Making the upperbound cardinality of some ECF messages unbounded, such as now allowing multiple CoreFilingMessage elements in a ReviewFilingRequest.

• Revising upperbound cardinality of some elements to unbounded.

• Adding an optional Country Codes enumeration.

However, in the CourtPolicyResponse, a new <RequireAsynchronousResponsesIndicator> element has been added. This element is mandatory. But GetPolicy is not mandatory for ECF. Although ECF implementations that do support ECF GetPolicy may want to invoke GetPolicy prior to ReviewFiling, GetPolicy is nevertheless a separate operation. Any backward compatibility issues with GetPolicy do not affect backward compatibility for ReviewFiling (provided that Court Policy is available to FAMDE).

So, if the ECF v4.01 review filing request was composed as recommended in ECF v4.01, i.e., consisting of a single CoreFilingMessage and an optional PaymentMessage, then theoretically, this review filing request could be understood by an ECF v4.1 ReviewFiling operation.

Namespace URI for ECF v4.01 namespaces are the same as the ECF v4.1 namespace URI, so no changes would be needed. Of course, if the ECF v4.01 implementation included implementation/court specific extensions, then court/implementation defined namespace URI may require modification (depending upon local standards and naming conventions).

Since the response XML for ReviewFiling has not been modified in ECF 4.1, there should not be any backward compatibility issues processing the response.

b. Can an ECF v4.01 request be sent to an ECF v4.1 operation using the ECF v4.01 Web Services SIP (i.e., **[ECF-v4.0-WS-SIP-v2.01]**)?

The short answer appears to be no. Even though wrappers.xsd is optional in ECF v4.1, it is not clear how ECF 4.1 would operate without it.

For example, the ReviewFiling operation invocation parameters specified in Section 5 of the Core Specification are CoreFilingMessage and (optionally) PaymentMessage. Although these parameters are provided in the request SOAP envelope, they are wrapped within a ReviewFilingRequest element which in turn is wrapped in a ReviewFiling SOAP Body child (e.g., exchange root) element.

So, whereas it appears that an ECF v4.1 CoreFilingMessage and PaymentMessage can be understood by an ECF v4.1 ReviewFiling operation, the ECF v4.01 WS-SIP (i.e., **[ECF-v4.0-WS-SIP-v2.01]**) could not be used to send it.

ECF has not provided a Web Services SIP that can be used to send ECF v4.01 exchanges to an ECF v4.1 implementation, but it is not too difficult to modify ECF v4.1 provided Web Services SIP v4.1 WSDL and wrappers.xsd to work with ECF v4.01. More information on this is provided in Appendix D.

Although it appears that there is a great deal of backward compatibility to ECF v4.01, there is not 100% backward compatibility. Additionally, it appears likely that an ECF v4.01 implementation may be able to communicate with an ECF v4.1 implementation using ECF v4.01 adapted Web Services ECF v4.1 as described in Appendix D.

### Backward Compatibility Summary

Since the combined ECF Core specification v4.1 and the corresponding Web Service SIP v4.1 define a complex, multi-faceted, e-filing eco-system, assessing backward compatibility is not a binary proposition. Different parts of ECF will have different backwards compatibility.

For the most part, a high degree of backward compatibility is supported. Most ECF messages created under ECF v4.01 should be successfully consumed by ECF v4.1 operations, provided the ECF messages can be delivered to the MDE operation. The exception is GetPolicyRequest due to the inclusion of <RequireAsynchronousResponsesIndicator>.

However, delivering ECF 4.01 Messages to ECF v4.1 MDE operations using Web Services is not backwardly compatible. An ECF v4.01 MDE would not be successful providing ECF v4.01 Messages to an ECF v4.1 MDE using the ECF v4.01 supported ECF Web Service-SIP v2.01. However, if the ECF v4.01 MDE used an ECF v4.01 adapted version of the ECF v4.1 Web Service SIP (as described in Appendix D), then this should result in successful delivery.

Nothing can be said regarding backward compatibility for court/local extensions. This must be evaluated by the implementor.

# Limitations, Caveats and Other Items of Note

This section identifies issues, understandings and other considerations that may be important for implementations to evaluate.

## Use of “Filing” Terminology

One should take note that the term “filing” is not used in a consistent manner within the ECF 4.1 (or prior) specifications.

As can be observed in the specification section cited above (i.e., Section 3.1.9 ‘GetFeesCalculation’), the term ‘filing’, in GetFeesCalculation refers to a CoreFilingMessage.

Take note that in the ECF Core specification, Section 1.4 ‘Terms and Definitions’, the term ‘filing’ is described as “an electronic document …”.

In the Core specification, Section 3.1 ‘The Filing-Preparation-to-Docketing Process Model’, the first paragraph ends with “other operations are optional and MAY occur within a given filing”. In this context, ‘filing’ must be understood, not as any single document, and also not as a single CoreFilingMessage, since multiple CoreFilingMessage(s) are allowed for a single submission (e.g., ReviewFilingRequest), but as a complete submission (that may contain multiple CoreFilingMessage(s) (for one or many cases) and multiple documents (both lead and connected)).

Section 3.2.4 ‘ReviewFiling’ says “the Filing Assembly MDE MUST submit the filing to the court by invoking the ReviewFiling operation”. In this context, ‘filing’ appears to mean ‘ReviewFilingRequest’. As of ECF 4.1, a ReviewFilingRequest may contain multiple CoreFilingMessage(s).

The RecordDocketingCallbackMessage (RDCM) includes the ecf:FilingStatus element. In ECF 4.01, a RDCM only allows one ReviewedLeadDocument (and zero to many child ReviewedConnectedDocument elements). Although the cardinality for RDCM within NDC has been modified in ECF 4.1 to permit unlimited ReviewedLeadDocument(s) within a single RDCM, the prior understanding for ‘filing’ still lingers, i.e., even in ECF 4.1, there remains a single FilingStatus element within a RDCM. Since there is only a single lead document in a RDCM, the inclusion of a single FilingStatus infers that ‘filing’ means ‘lead document and its connected documents’. As such, when an RvFR provides a CoreFilingMessage with multiple Lead Documents, from the RDCM FilingStatus perspective, this is multiple ‘filings’.

In Section 3.2.5 ‘ServeFiling’, it states “the Filing Assembly MDE MAY serve the entire filing, to other parties in the case by invoking the ServeFiling operation on the ServiceMDE”. In this context, the term ‘filing’ should be understood as a single CoreFilingMessage (including the documents contained or referenced therein) from within a ReviewFilingRequest.

Overall, when the term ‘filing’ appears within the specification, it often refers to either a single CoreFilingMessage (that may provide one or many documents, lead and connected) and sometimes refers to a complete ReviewFilingRequest (that may provide many CoreFilingMessage elements). In at least one instance (e.g., FilingStatus) ‘filing’ refers to a single lead document and its connected documents.

The term ‘filing’ rarely means ‘an electronic document’ as stated in the ‘Terms and Definitions’ section.

Consider FilingStatusResponseMessage. This message is restricted to a single nc:Case (or case type specific element via substitution), a single nc:DocumentIdentification, and a single ecf:FilingStatus. This structure does not lend itself to supporting multiple CoreFilingMessage(s) within a single ‘filing’ (e.g., ’ReviewFilingRequest’), especially when multiple cases are provided within a single ‘filing’.

## Filing Identifier

Also consider “filing identifier” (e.g., Section 3.3.1.5). It seems reasonable that single (and locally (e.g., court) unique) “filing identifier” is assigned to a single ‘filing’.

The specification language provided in Section 3.3.1.5 ‘Filing Identifiers’ suggests that a single ‘filing identifier’ is assigned for a single ReviewFilingRequest (e.g., “will be generated by the court in response to a ReviewFiling operation”). This single ‘filing identifier’ to a single ReviewFilingRequest is also implied within Section 3.2.4 ‘ReviewFiling’, as: “The Filing Review MDE responds synchronously with a receipt message that includes the filing identifier issued by the court.” (singular).

More practical is that a unique ‘filing identifier’ would be assigned for each CoreFilingMessage. As such, when a single ReviewFilingRequest provides multiple CoreFilingMessage(s), then multiple ‘filing identifiers’ should be assigned (i.e., one for each CoreFilingMessage). In this context ‘filing’ (as designated by a ‘filing identifier’) refers to a CoreFilingMessage.

The Core specification does not specify how ‘filing identifiers’ are specified (e.g., which element(s) to use, etc.), for contrast see Section 3.3.1.8 ‘Filer and Party Identifiers’ in which a non-normative example is provided).

The Core specification does not establish where, when or how filing identifiers are generated. The implication from Section 3.2.4, quoted above, is that assignment, and perhaps generation, of filing identifiers is done by the FRMDE. Nevertheless, it does seem clear that the filing identifiers are expected to be provided to the FAMDE in the ReviewFilingResponse within MessageReceiptMessage. Since this response is synchronous, then filing identifiers must either be generated in the FRMDE or pre-generated then assigned by the FRMDE, and all filing identifiers assigned for a single ReviewFilingRequest must be returned together.

## Multiple CoreFilingMessages in ReviewFilingRequest

The ReviewFilingRequest may now include more than one CoreFilingMessage. There are few restrictions imposed on this capability. Implementors are cautioned about exploiting this feature.

Although not specifically called out in the Core specification, one limitation is that all CoreFilingMessage(s) should be destined to the same court, even though the ReviewFilingRequest structures and schema would not prevent multiple courts. This single court destination is implied by the Core specification when it states (Section 3.2.4 ‘ReviewFiling’) “The Filing Assembly MDE MUST submit the filing to the court”. In this context, ‘filing’ would refer to a single ReviewFilingRequest. The use of the term “the court” implies a single court.

Mixed cases within a single ReviewFilingRequest are also not precluded, provided all the cases are at the same court. Since only a single PaymentMessage is allowed within the ReviewFilingRequest for the multiple mixed-case CoreFilingMessage(s), this may provide natural governance limiting the mix of cases.

Multiple filing-identifiers are recommended, one for each CoreFilingMessage. Filing-identifiers are to be provided to the FAMDE in the ReviewFiling response. The Core specification does not identify how this would be done within a single ReviewFiling response. In fact, the Core specification does not even identify how a single filing-identifier should be returned in a MessageReceiptMessage to the FAMDE from the FRMDE.

The Core specification does not preclude rejection of submissions prior to clerk review (e.g., by an EFM). Rejection could be due to not well-formed XML or XML not valid to schema, or other cause, such as invalid values, documents exceeding size limits, corrupt documents, viruses, etc.

When a submission provides a single CoreFilingMessage, then communicating this rejection status is simpler since the MessageReceiptMessage returned is addressing a single CoreFilingMessage.

However, when a ReviewFilingRequest provides more than one CoreFilingMessage, then the use of a single MessageReceiptMessage is more challenging. Although MessageReceiptMessage permits multiple errors to be reported, ECF does not provide any mechanism for associating specific errors to specific CoreFilingMessage(s). At present, ECF provides no guidance for this circumstance.

Additionally, the Core specification does not preclude implementations from splitting up ReviewFilingRequest(s) following submission by the FAMDE. For example, a single ReviewFilingRequest that contains multiple CoreFilingMessage(s) could be divided into multiple exchange units by an EFM (e.g., one exchange for each CoreFilingMessage), before being provided to the FRMDE (e.g., clerk review). In this circumstance, multiple ‘filing identifiers’ would be especially useful.

When an EFM rejects one or more, but not all CoreFilingMessage(s) within a single ReviewFilingRequest, and then sends the un-rejected CoreFilingMessage(s) on to the FRMDE, then any MessageReceiptMessage returned to the FAMDE should be clear and specific when identifying the rejected CoreFilingMessage(s). ECF does not provide any guidance on this within the Core specification.

One approach that implementations may consider, would be to leverage the nc:DocumentIdentification element within MessageReceiptMessage (MRM), using this element to list all CoreFilingMessage(s) to which the MRM applies (e.g., by providing the CoreFilingMessage’s filing identifier in nc:DocumentIdentification/nc:IdentificationID, presuming that one filing-identifier is assigned to each CoreFilingMessage). So, if the ReviewFilingRequest provides 8 CoreFilingMessage(s), and each CoreFilingMessage is provided a unique filing-identifier, then the response MessageReceiptMessage would include 8 instances of nc:DocumentIdentification, with each instance returning a unique filing-identifier.

By itself, the above proposal still does not allow a specific ecf:Error to be associated with a specific CoreFilingMessage. This linkage may perhaps be provided by utilizing attributes (e.g., s:id, s:metadata, and/or s:linkMetadata).

When a ReviewFilingRequest provides multiple CoreFilingMessage(s), there are implications for clerk review (e.g., multi-episode clerk review, etc.) and for RecordFilingRequest(s).

#### Multi-Episode Clerk Review with multiple CoreFilingMessages

Multi-Episode Clerk Review occurs when a full ReviewFilingRequest is reviewed in more than one clerk review session and when submission documents and/or CoreFilingMessage(s) are accepted or rejected (or other disposition) in each session without accepting or rejecting all submission documents within a single session, and when the intermediate results are forwarded to the next operation (i.e., RecordFiling). As such, it takes two or more sessions to provide clerk review dispositions for all documents within a submission. These sessions may be minutes, hours or even days apart.

Multi-Episode clerk review is possible even when a ReviewFilingRequest contains a single CoreFilingMessage. It may have greater probability when the ReviewFilingRequest contains multiple CoreFilingMessage(s).

When a ReviewFilingRequest contains a single CoreFilingMessage, multi-episode clerk review is possible when the CoreFilingMessage provides multiple filing documents, especially multiple FilingLeadDocument(s). Generally, when a FilingLeadDocument is either accepted or rejected all of its FilingConnectedDocument(s) are also either accepted or rejected (e.g., a FilingLeadDocument and its FilingConnectedDocument(s) are generally reviewed as a unit).

When multi-episode clerk review occurs, then as a result, there will be multiple RecordFiling operation requests for a single ReviewFilingRequest (when the ECF RecordFiling operation is used).

#### RecordFilingRequests for multiple CoreFilingMessages

Even when a submission (e.g., ReviewFilingRequest) contains a single CoreFilingMessage, there may be multiple RecordDocketingCallbackMessage(s). In a fully reviewed ReviewFilingRequest, there would be one RecordDocketingCallbackMessage for each FilingLeadDocument in ECF v4.01, but now with ECF v4.1, multiple ReviewedLeadDocument(s) may be accommodated within a single RecordDocketingCallbackMessage.

When the ReviewFilingRequest contains multiple CoreFilingMessage(s) then there are several possible permutations.

The first, and simplest permutation would be to provide a single RecordFilingRequest to the RecordFiling operation that contained all of the ReviewFilingRequest’s provided CoreFilingMessage(s), and also contained one, and only one, corresponding RecordDocketingCallbackMessage for each CoreFilingMessage. For this simplest, most straight forward option, each RecordDocketingCallbackMessage would contain a ReviewedLeadDocument element for each FilingLeadDocument in its corresponding CoreFilingMessage.

## Service List Registry

Court service list registry – Section 3.2.2 ‘GetServiceInformation’ identifies a “Court’s registry” as the source for case participant service information (e.g., contact information). The concept of “court registry” is not defined in the specification. As such, Core specification mandates, such as “there MUST be only one such registry per court” and “the Court Record MDE MUST have access to the court’s registry”, are unclear (and perhaps also unenforceable, as in not required for a complete, or compliant implementation).

Also, keep in mind that GetServiceInformation is an optional operation that implementations need not support.

## Hub Service

Hub Service – this type of service implementation is raised in Section 3.2.5 ‘ServeFiling’. ‘Hub Service’ also appears in Section 3.2.2 ‘GetServiceInformation’.

A ‘Hub Service MDE’ is defined in Section 1.4 ‘Terms and Conditions’ as “A centralized Service MDE capable of receiving a single set of service notifications for all parties registered for electronic service in a case and transmitting the service notifications to the Service MDEs registered to each party in the case.”

A Hub Service is not a true MDE and instead is a proxy for one or more other MDE implementations. The Core specification only considers ‘Hub Services’ for the Service MDE, specifically ServeFiling and GetServiceInformation.

Other possible uses for ‘Hub Services’ could be considered. For example, a centralized Electronic Filing Manager (EFM) could service multiple FRMDEs, e.g., one or more in one or many courts. This same Hub Service could also handle all GetPolicy requests for all supported CRMDEs.

## Implementation Namespace

Web Services SIP v4.1 provides 4 example implementation WSDL in addition to the four (4) specification provided, normative base WSDL, e.g., one for each MDE. Implementation WSDL are expected to import the specification provided MDE base WSDL. As shown in the provided example implementation WSDL, each implementation WSDL is expected to define an implementation specific namespace.

The ECF specifications do not provide guidelines for these namespace names. The implementation examples all provide the same namespace name:

urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:WebServices-ImplementationExample-4.1

TC: Should separate namespace names be recommended (e.g., should implementations define four different namespaces, one for each MDE, or should implementations define a single namespace which encompasses all four MDE)?

## Payment Maximum Amount

In Section 3.3.3.2 ‘PaymentMessage’, the specification states “The payment MAY include a maximum amount for the payment if some latitude is needed to accomplish the filing.” The specification does not identify the element or elements used for this purpose (e.g., ecf:MaximumAmount). There is no obvious element (based on element names and descriptions) for this maximum amount.

Perhaps the specification statement is intended to suggest that courts/implementations are free to impose the largest allowed fee(s).

Additionally, there does not appear to be any support in GetFeesCalculation, either in the request or in the response, for a payment maximum amount.

TC: Perhaps the ECF TC would like to clarify this provision.

## ECF 5 Alignment

Although many of the changes that appear in ECF v4.1 are intended to better align with ECF 5.x (especially ECF 5.01), some important differences remain. These include:

a. No new operations, optional or otherwise, have been included in ECF 4.1. ECF 5.x introduces several new operations, i.e., CancelFiling, DocumentStampInformation, GetCourtSchedule, RequestCourtDate, ReserveCourtDate, AllocateCourtDate, and NotifyCourtDate.

b. Case type specific elements in ECF 4.1 are still only intended for use for case initiation filing submissions and are not intended for use with subsequent filing submissions.

This distinction is most noticeable in that with ECF 5.x the nc:Case element may include ecf:CaseAugmentation and j:CaseAugmentation, whereas with ECF 4.x, nc:Case does not provide any augmentation elements.

The ECF TC understands that some or many ECF 4.x implementations may also use case type specific elements within subsequent filing submissions.

## NIEM Version

The version of NIEM has not been modified for ECF v4.1

Updating the NIEM version from MIEM 2.0 to NIEM 2.1 was considered for ECF 4.1. Minor version number updates in NIEM are supposed to be backward compatible.

However, when investigating this, it was discovered that the element j:StatuteOffenseIdentification had been removed from j:StatuteType in NIEM 2.1.

To preserve the greatest degree of backward compatibility, it was decided to stay with NIEM 2.0 for ECF 4.1.

Since the same version of NIEM is used, this means that any code-lists used by ECF from NIEM will have the very same code list values as available in ECF v4.01. This may have implications on newer implementations.

For example, if new countries have been added or former countries have vanished or country names have been changed, then the older NIEM code lists (e.g., ISO 3166) will not contain these new or updated values and will continue to include obsolete values.

To mitigate this limitation, future versions of ECF, such as ECF 5 will rely more heavily on Genericode (gc) code lists and less so on schema enumerated code lists.

## ECF Conformance and Compliance

ECF does not provide OASIS recommended conformance clauses, relying instead on a blanket statement in section 8 ‘Conformance’ in the Core specification:

“*An implementation conforms with the Electronic Court Filing Version 4.1 if the implementation meets the requirements in Sections 1-6 including conformance with the XSD schemas and* **[Genericode]** *code lists* referenced in Section 3 and 4”

The TC may consider writing specific numbered conformance clauses in the future, as required by OASIS (see [docs.oasis-open.org/templates/TCHandbook/ConformanceGuidelines.html](https://docs.oasis-open.org/templates/TCHandbook/ConformanceGuidelines.html)).

In addition to the blanket conformance statement provided in the Core specification, the following additional compliance statements are included:

Section 2.2 ‘Major Design Elements’ of the Core specification provides: “An ECF 4.1-compliant implementation may implement one or more of the MDEs defined in the specification but a complete ECF 4.1 system MUST include at least one each of the Filing Assembly, Filing Review and Court Record MDEs.”

The words “one each” imply that, for any given MDE, there may be multiple components that provide the operations implemented for that MDE. These words should not be understood as multiple MDEs, such as multiple FAMDEs within an implementation. Instead, there would be a single FAMDE, but there may be multiple FAMDE service providers. The Section 2.2 statement cited above is further illustrated with “For instance, a court may decide to provide certain MDEs and allow private providers to furnish the remaining MDEs”.

This section also provides” In order to be compliant with ECF 4.1, an MDE MUST support all required operations for that MDE. However, in an ECF 4.1 system that does not support electronic service, the operations associated with the Legal Service MDE are not required”.

Additionally, section 3.1 ‘The Filing-Preparation-to-Docketing Process Model’ provides “The ReviewFiling and RecordFiling operations are required in a complete ECF 4.1 system as prescribed in Section 2.2. However, when the RecordFiling operation has been implemented within the same system as the ReviewFiling operation, then the RecordFiling operation need not be provided in an ECF 4.1 compliant manner”.

Section 2.2 ‘Core vs. Profiles’ provides “In order to be compliant, an implementation of the ECF specification MUST implement the core specification and at least one service interaction profile and one document signature profile”.

Section 2.4.1 ‘Human-Readable Court Policy’ states “To be compliant with the ECF 4.1 specification, each court MUST publish a human-readable court policy that MUST include each of the following” and then provides a listing of information that must be included.

In the above, three terms are used: 1) conformant, 2) compliant, and 3) complete.

It appears that an ‘implementation’ can be ‘complete’ and not also be either ‘conformant’ or ‘compliant’.

To be ‘complete’, an ‘implementation’ must:

* Provide the ReviewFiling operation, presumably in an ECF 4.1 compliant manner, and
* Provide the RecordFiling operation, however the RecordFiling operation need not be ECF 4.1 compliant when RecordFiling and ReviewFiling are within the same system, and
* Include the Filing Assembly MDE, the Filing Review MDE and the Court Record MDE.

To be ‘compliant’, an ‘implementation’ must:

* Implement a Service Interaction Profile (e.g., Web Services SIP v4.1), and
* Implement at least one document signature profile, and
* Publish human-readable court policy as required in section 2.4.1 of the Core specification, and
* Implement at least one MDE, but may implement more than one MDE, and
* Provide all required operations for implemented MDEs.

To be ‘conformant’, an ‘implementation’ must:

* Be a ‘compliant’ implementation, and
* Meet the requirements stated in sections 1 – 6 in the Core specification, and
* Conform with ECF v4.1 schema, and
* Conform with Genericode lists provided in sections 3 and 4 in the Core specification.

It is not clear what it means to conform with the Genericode lists. Section 3.3.2 ‘Code Lists’ provides a listing of normative code lists. Some of the included code lists are Genericode lists. ECF v4.1 provides 16 Genericode lists in the ‘gc’ folder. Each of these Genericode lists contain lists of codes.

It is not clear whether Genericode compliance requires the use of all 16 Genericode lists (when applicable) as provided by ECF v4.1 or whether the code lists can be extended or contracted.

The statement “when applicable” as used above, means that the Genericode list is used when the exchange and/or case type includes elements controlled by the code list. Determining this applicability, is in itself, tenuous. There are no normative statements in the ECF v4.1 specification that associate specific elements with specific Genericode lists, as is done with schema enumerated code lists. It appears that the association is ‘suggested’ by Genericode filename, e.g., the Genericode file ‘ECF-4.1-FilingStatusCode.gc’ governs the contents of the ‘FilingStatusCode’ element (fortunately this element name only appears in one namespace). Some ECF elements provide non-normative documentation in the element definition, provided in schema, that may aid in this association. For example, the element description for ecf:FilingStatusCode is “Status of the filing as determined by the system sending the callback. Allowable values defined in ECF-4.0-FilingStatusCode.gc.” Clearly the reference should be to ‘ECF-4.1-FilingStatusCode.gc’ rather than to ECF-4.0.

To aid in the determination whether all requirements stated in Core specification sections 1 - 6 have been met, and in the absence of conformance clauses, a listing of specification normative statements is included in Appendix F.

1. Informative References

This appendix contains the references that are used in this document.

While any hyperlinks included in this appendix were valid at the time of publication, OASIS cannot guarantee their long-term validity.

Note: Each reference to a separate document or artifact in this work must be listed here.

Recommended approach: Set up **[Citation]** label elements as "Bookmarks" or HTML tags, then create hyperlinks to them within the document.

If there is a hyperlink, it should be visible and linked.

The proper format for citation of technical work produced by an OASIS TC (whether Standards Track or Non-Standards Track) is:

**[Citation Label]**

Work Product title (italicized). Edited by Albert Alston, Bob Ballston, and Calvin Carlson. Approval date (DD Month YYYY). OASIS Stage Identifier and Revision Number (e.g., OASIS Committee Specification Draft 01). Principal URI (version-specific URI, e.g., with stage component: somespec-v1.0-csd01.html). Latest version: (static URI, without stage identifiers).

For example:

**[OpenDoc-1.2]**

Open Document Format for Office Applications (OpenDocument) Version 1.2. Edited by Patrick Durusau and Michael Brauer. 19 January 2011. OASIS Committee Specification Draft 07. https://docs.oasis-open.org/office/v1.2/csd07/OpenDocument-v1.2-csd07.html. Latest version: https://docs.oasis-open.org/office/v1.2/OpenDocument-v1.2.html.

Reference sources:

For references to IETF RFCs, use the approved citation formats at:

https://docs.oasis-open.org/templates/ietf-rfc-list/ietf-rfc-list.html.

For references to W3C Recommendations, use the approved citation formats at:

https://docs.oasis-open.org/templates/w3c-recommendations-list/w3c-recommendations-list.html.

Remove this note before submitting for publication.

[RFC3552]

Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <<https://www.rfc-editor.org/info/rfc3552>>.

[Reference]

[Full reference citation]

1. Acknowledgments

[Required section]

Note: A Work Product approved by the TC must include a list of people who participated in the development of the Work Product. This is generally done by collecting the list of names in this appendix. This list shall be initially compiled by the Chair, and any Member of the TC may add or remove their names from the list by request. Remove this note before submitting for publication.

The following individuals have participated in the creation of this committee note and are gratefully acknowledged:

Participants:

Gary Graham, Arizona Supreme Court

 Add names of additional contributors.

* 1. Special Thanks

This is an optional subsection to call out contributions from TC members. If a TC wants to thank non-TC members then they should avoid using the term "contribution" and instead thank them for their "expertise" or "assistance".

* 1. Participants

A TC can determine who they list here, however, Observers must not be listed. It is common practice for TCs to list everyone that was part of the TC during the creation of the document, but this is ultimately a TC decision on who they want to list and not list.

The following individuals were voting members of this Technical Committee during the creation of this document and their contributions are gratefully acknowledged:

Philip Baughman, Tyler Technologies, Inc.

James Cabral, InfoTrack US

Eric Eastman, InfoTrack US

Gary Graham, Arizona Supreme Court

Ryan Foley, ImageSoft, Inc.

George Knecht, InfoTrack US

Mark Leong, Arizona Supreme Court

James, McMillan, National Center for State Courts

Enrique Othon, Tyler Technologies, Inc.

Jim Price, Arizona Supreme Court

Brock Rogers, File & ServeXpress

Patrick Wallace, Tyler Technologies, Inc.

1. Wrappers.xsd structures

The table below lists all type-structures defined within wrappers.xsd. For each type-structure, the element derived from this structure is provided in the middle column, i.e., “Derived Element”. Often this derived element is the operation invocation parameter, named as required in the ECF v4.1 WS-SIP (e.g., Section 2.5 ‘Request and Operation Invocation’; named using the operation name). The rightmost column provides the name of the invocation parameter. Request rows with ‘N/A’ (not applicable) in the rightmost column identify request Type Structures that are used to derive elements that in turn are used to derive additional Type Structures. The rightmost column is grey for response types, since responses are never used as invocation parameters.

To illustrate, the element ‘NotifyFilingReviewCompleteRequest’ (derived from ‘NotifyFilingReviewCompleteRequestType’) is the exclusive content of ‘NotifyFilingReviewCompleteType’ (from which ‘NotifyFilingReviewComplete’ has been derived). When the NotifyFilingReviewComplete operation is invoked, it uses the NotifyFilingReviewComplete element as the invocation parameter.

|  |  |  |
| --- | --- | --- |
| **Type Structure** | **Derived Element** | **WSSIP Operation Invocation** |
| GetPolicyRequestType | GetPolicy | GetPolicy |
| GetCaseListRequestType | GetCaseList | GetCaseList |
| GetCaseRequestType | GetCase | GetCase |
| GetDocumentType | GetDocument | GetDocument |
| GetFeesCalculatonRequestType | GetFeesCalculation | GetFeesCalculation |
| GetFilingListRequestType | GetFilingList | GetFilingList |
| GetFilingStatusRequestType | GetFilingStatus | GetFilingStatus |
| GetServiceInformationRequestType | GetServiceInformation | GetServiceInformation |
| ServeFilingRequestType | ServeFiling | ServeFiling |
| NotifyDocketingCompleteRequestType | NotifyDocketingComplete | NotifyDocketingComplete |
| NotifyFilingReviewCompleteRequestType | NotifyFilingReviewCompleteRequest | N/A |
| NotifyFilingReviewCompleteType | NotifyFilingReviewComplete | NotifyFilingReviewComplete |
| RecordFilingRequestType | RecordFilingRequest | N/A |
| RecordFilingType | RecordFiling | RecordFiling |
| ReviewFilingRequestType | ReviewFilingRequest | N/A |
| ReviewFilingType | ReviewFiling | ReviewFiling |
| GetPolicyResponseType | GetPolicyResponse |  |
| GetCaseListResponseType | GetCaseListResponse |  |
| GetCaseResponseType | GetCaseResponse |  |
| GetDocumentResponseType | GetDocumentResponse |  |
| GetFeesCalculationResponseType | GetFeesCalculationResponse |  |
| GetFilingListResponseType | GetFilingListResponse |  |
| GetFilingStatusResponseType | GetFilingStatusResponse |  |
| GetServiceInformationResponseType | GetServiceInformationResponse |  |
| ServeFilingResponseType | ServeFilingResponse |  |
| NotifyDocketingCompleteResponseType | NotifyDocketingCompleteResponse |  |
| NotifyFilingReviewCompleteResponseType | NotifyFilingReviewCompleteResponse |  |
| RecordFilingResponseType | RecordFilingResponse |  |
| ReviewFilingResponseType | ReviewFilingResponse |  |

In the table above, names highlighted with yellow are request type structures that do not include ‘Request’ within the type name. Names highlighted in blue are type structures and elements that do not result in an invocation parameter named element.

1. Adapting ECF v4.1 Web Services SIP to ECF 4.01
	1. Summary

This section explores the use of the ECF 4.1 Web Services SIP with ECF v4.01.

The approach considered here will be to use the ECF 4.1 WS SIP four MDE specific WSDL, minimally modified as necessary, to work with ECF v4.01. To do this, the new ECF v4.1 wrappers.xsd will need to be used with ECF v4.01. Wrappers.xsd will require minimal modification for use in ECF v4.01.

* 1. Introduction

As originally conceived, SIPs are independent of the main ECF specification, permitting implementers to choose one or more SIPs from a library of available SIPs. This library has always been small, consisting of the Portable Media SIP, Web Services SIP and the committee draft IBM MQ SIP. There have been recent conversations about other possible SIPS such as a REST/XML SIP (see ECF TC F2F Meeting Minutes for 12-07-2022), and once an Email SIP had been considered.

Although ECF implementers may choose from a library of SIPs (one or multiple) for an implementation, mixing and matching has not been intended to operate across different versions.

For example, the Introduction for ECF 4.0 Web Services Service Interaction Profile Version 2.01 states that it is “for use with [ECF 4.0] specification” and also provides section 1.1 “Relationship to ECF 4.0 Specifications”. The reference to ECF 4.0 is understood as ECF v4.0 or its minor variants, such as ECF v4.01, and not to the whole family of ECF 4.x specifications (including ECF 4.1).

Similarly, the Introduction in ECF 4.1 Web Services Service Interaction Profile states, “for use with the [ECF 4.1] specification” and also provides section 1.1 “Relationship to ECF 4.1 Specifications”. This specification also states that the new specification replaces or supersedes Web Services Interaction Profile 2.01 Specification.

* 1. Rationale

Can the Web Services SIP 4.1 be used successfully with ECF 4.01?

First off, the new WebServices SIP 4.1 is much cleaner and more practical than the older v2.0 and v2.1 SIPs, so the above question is not just academic, it also has practical value.

The existing Web Services SIPs for ECF v4.01 have difficulties long considered by the ECF TC. These include:

a. A gap exists between the ECF v4.01 and its two committee recommended Web Services SIPs (i.e., v2.0 and v2.1). This gap must be filled by implementations through the use of implementation specific exchange schema. When using version adapted wrappers.xsd and WS SIP v4.1, no exchange schema gaps will exist.

b. The WS SIPs for ECF v4.01 specify, in Section 2.5 “Request and Operation invocation”,

Each message transmission MUST identify the operation being invoked within the SOAP Body only; the (qualified) operation name MUST be the qualified name of the first child element of the SOAP body element, as called for in section 7.1 of the [SOAP 1.1] specification.

Whereas WS SIP v4.1 conforms to the requirement within Section 2.5, older versions of Web Services SIP do not.

c. The issue described above was raised during the ICJIS Springboard project (see <https://www.oasis-open.org/apps/org/workgroup/legalxml-courtfiling/download.php/54588/ECF%20Springboard%20Quality%20Assurance%20Review%20v.1.0.1.docx>) and was never resolved.

d. Given the issue identified above, many ECF 4 implementers have followed an implementation implied by the ECF Web Services SIP provided WSDL and therefore may not have provided Section 2.5 conformant implementation specific exchange schema. This results in SOAP Body contained XML that is not conformant the WSDL xsd and relevant schema.

A non-normative conformant example (D.1) is illustrated in Appendix D of the WS SIP specification document.

This example is copied below, with yellow highlighting added:

MIME-Version: 1.0

Content-Type: Multipart/Related; boundary=boundary;

type=”application/xop+xml”;

 start="Envelope"

start-info=”text/xml”

--boundary

Content-Type:application/xop+xml;

 text/xml; charset="UTF-8"

Content-Transfer-Encoding: 8bit

Content-ID: Envelope

<?xml version='1.0' ?>

<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">

 <env:Body xmlns:types="http://example.com/some-namespace">

 <types:ReviewFiling>

 <CoreFilingMessage>

 …

 </CoreFilingMessage>

 <PaymentMessage>

 …

 </PaymentMessage>

 </types:ReviewFiling>

 </env:Body>

</env:Envelope>

The example shows compliance with section 2.5 by providing <types:ReviewFiling> as the immediate child element beneath the SOAP <env:Body> element.

However, in terms of ‘wsdl alignment’, this is not the resultant SOAP when a SOAP request is generated from specification provided WSDL. This can be verified using either XMLSpy or SoapUI.

* 1. Considerations

The language in the Web Services SIP for ECF-4.01 appears to require the use of ECF-4.0-WebServicesProfile-Definitions.wsdl, however nothing within the specification suggests that modifications to this wsdl are prohibited. Even if properly interpreted, since WS SIP v4.1 will be used instead of WS SIP v2.01, ECF-4.0-WebServicesProfile-Definitions.wsdl will not be used.

However, the ECF v4.01 specification, in Section 5.2 “Service Interaction Profile Approval and Revision Process” states that “ECF Technical Committee (TC) will recommend certain service interaction profiles for use in implementations of the ECF 4.0 specification.” This statement suggests that successful results of this exploration should be submitted as a new service interaction profile and not just as a committee note.

When an ECF v4.01 MDE sends an exchange (e.g., ECF v4.01 RecordFilingRequest) to an ECF v4.1 MDE, using an ECF v4.01 adaptation of the ECF WS-SIP v4.1, as described in this Appendix, then Service Interaction Profile Identifier, provided as SendingMDEProfileCode, should be the WS-SIP v4.1 identifier, i.e.:

urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:WebServices-4.1

Compare and Contrast:

ECF 4.1 provides the new wrappers.xsd. This wrappers.xsd is required for use with Web Services 4.1 and will need to be incorporated into ECF v4.01. Modifications will need to be made to wrappers.xsd to accommodate ECF v4.01. Depending upon XML environments and folders, the ECF v4.01 adaptation of wrappers.xsd may not need to be named using a different filename (e.g., may be able to be named as ‘wrappers.xsd’ and not be named differently, such as ‘ECF 4.01 Wrappers.xsd’).

ECF 4.1 Web Services SIP is divided into 4 separate WSDLs, one for each MDE. Prior Web Services SIPs provided a single WSDL for all MDEs. Depending upon XML environments, WSDL filenames may be the same as the ECF v4.1 source files, or may need to be named differently.

Approach:

1. Copy wrappers.xsd from ECF v4.1 and modify for ECF v4.01.

2. Copy 4 MDE specific WSDL from WS SIP v4.1 and modify for v4.01.

3. Copy/create implementation MDE specific WSDL.

* 1. Artifact Modifications

**Wrappers.xsd modifications for ECF v4.01:**

1. In <schema> element, modify all namespace URI for ECF namespaces, replacing ‘4.1’ with ‘4.0’.

For example,

xmlns:docketcb="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:RecordDocketingCallbackMessage-4.1"

Must be revised, replacing ‘4.1’ with ‘4.0’

2. Modify the targetNamespace URI, replacing ‘4.1’ with ‘4.0’.

For example,

targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1"

Must be revised, replacing ‘4.1’ with ‘4.0’

3. Modify the value for the version attribute, setting it to “4.0” instead of “4.1”.

For example,

version="4.1"

Must be revised, replacing ‘4.1’ with ‘4.0’

4. Within ‘imports’, modify both namespace URI and schemaLocation, replacing ‘4.1’ with ‘4.0’.

For example:

<xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ReviewFilingCallbackMessage-4.1" schemaLocation="message/ECF-4.1-ReviewFilingCallbackMessage.xsd"/>

Must be revised, replacing ‘4.1’ with ‘4.0’

**WSDL modifications for ECF 4.01:**

In each of the four MDE specific WSDL:

1. Attribute values within the <definitions> element require modification to replace the version number ‘4.1’ within namespace URI with ‘4.0’. This applies to the ‘targetNamespace’ attribute, the ‘xmlns:tns’ attribute and the ‘xmlns:wrappers’ attribute.

For example:

<definitions targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.1"

 xmlns:tns="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.1"

 xmlns:wrappers="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"

 xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"

 xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200702"

 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">

Must be revised, replacing ‘4.1’ with ‘4.0’

3. Within soapAction for each operation (e.g., “NotifyFilingReviewComplete”), replace ‘4.1’ with ‘4.0’.

For example:

<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>

<operation name="NotifyFilingReviewComplete">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfilingschema:wsdl:FilingAssemblyMDE-4.1\NotifyFilingReviewComplete"/>

 <input>

 <soap:body use="literal"/>

 </input>

Must be revised, replacing ‘4.1’ with ‘4.0’

In addition to the four MDE specific WSDL provided in WS-SIP v4.1, there are 4 implementation examples WSDL also provided. The import element in implementation MDE WSDL will also need to be modified to replace “4.1” with “4.0”:

<import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.1" location="../FilingAssemblyMDE.wsdl"/>

Must be revised, replacing ‘4.1’ with ‘4.0’

* 1. Examples

Example 1 - ECF 4.01 adapted wrappers.xsd

<?xml version="1.0" encoding="UTF-8"?>

<!--

 Wrappers.xsd

 This version of wrappers.xsd has been modified from the 4.1 version and adapted for ECF 4.0.

-->

<!--

 Modifications to make Wrappers.xsd valid in an ECF 4.01 environment (12-16-2022):

 Modified the 4.1 version number in namespace URI within the namespace prefix declarations in

 the schema element, replacing '4.1' with '4.0'.

 Modified the 4.1 version number within the import elements for both namespace URI and

 schemaLocation, replacing '4.1' with '4.0'.

-->

<!--

 LegalXML Electronic Court Filing Version 4.1

 Copyright (c) OASIS Open 2022. All Rights Reserved.

 Source: http://docs.oasis-open.org/legalxml-courtfiling/ecf/v4.1

-->

<!--

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

-->

<!--

 This XSD document defines the types and elements for each operation on all Major Design Elements (MDEs) in the OASIS Electronic Court Filing 4.0 specification.

 Oct 2015: Changes applied to all request messages to align with operations named in Section 2.5 of the Web Services Service Interaction Profile, including new types added for document/literal wrapped patterns.

 Mar 2016: Extracted type definitions from WSDL into this document and added import statement to WSDL definition document.

-->

<xsd:schema xmlns="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:nc="http://niem.gov/niem/niem-core/2.0" xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200702" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-361 wssecurity-utility-1.0.xsd" xmlns:i="http://niem.gov/niem/appinfo/2.0" xmlns:caselistquery="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseListQueryMessage-4.0" xmlns:caselistresponse="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseListResponseMessage-4.0" xmlns:casequery="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseQueryMessage-4.0" xmlns:caseresponse="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseResponseMessage-4.0" xmlns:casetype="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseTypeSpecificMessage-4.0" xmlns:core="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CoreFilingMessage-4.0" xmlns:policyquery="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CourtPolicyQueryMessage-4.0" xmlns:policyresponse="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CourtPolicyResponseMessage-4.0" xmlns:court="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CourtSpecificMessage-4.0" xmlns:docquery="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:DocumentQueryMessage-4.0" xmlns:docresponse="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:DocumentResponseMessage-4.0" xmlns:ecf="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CommonTypes-4.0" xmlns:feesquery="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FeesCalculationQueryMessage-4.0" xmlns:feesresponse="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FeesCalculationResponseMessage-4.0" xmlns:listquery="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FilingListQueryMessage-4.0" xmlns:listresponse="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FilingListResponseMessage-4.0" xmlns:statusquery="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FilingStatusQueryMessage-4.0" xmlns:statusresponse="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FilingStatusResponseMessage-4.0" xmlns:message="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageReceiptMessage-4.0" xmlns:payment="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:PaymentMessage-4.0" xmlns:receipt="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:PaymentReceiptMessage-4.0" xmlns:docketcb="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:RecordDocketingCallbackMessage-4.0" xmlns:docket="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:RecordDocketingMessage-4.0" xmlns:reviewcb="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ReviewFilingCallbackMessage-4.0" xmlns:servicequery="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ServiceInformationQueryMessage-4.0" xmlns:serviceresponse="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ServiceInformationResponseMessage-4.0" xmlns:servicereceipt="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ServiceReceiptMessage-4.0" targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0" version="4.0">

 <xsd:import namespace="http://niem.gov/niem/niem-core/2.0" schemaLocation="constraint/niem/niem-core/2.0/niem-core.xsd"/>

 <xsd:import namespace="http://niem.gov/niem/appinfo/2.0" schemaLocation="constraint/niem/appinfo/2.0/appinfo.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseListQueryMessage-4.0" schemaLocation="message/ECF-4.0-CaseListQueryMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseListResponseMessage-4.0" schemaLocation="message/ECF-4.0-CaseListResponseMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseQueryMessage-4.0" schemaLocation="message/ECF-4.0-CaseQueryMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseResponseMessage-4.0" schemaLocation="message/ECF-4.0-CaseResponseMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CoreFilingMessage-4.0" schemaLocation="message/ECF-4.0-CoreFilingMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CourtPolicyQueryMessage-4.0" schemaLocation="message/ECF-4.0-CourtPolicyQueryMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CourtPolicyResponseMessage-4.0" schemaLocation="message/ECF-4.0-CourtPolicyResponseMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:DocumentQueryMessage-4.0" schemaLocation="message/ECF-4.0-DocumentQueryMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:DocumentResponseMessage-4.0" schemaLocation="message/ECF-4.0-DocumentResponseMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FeesCalculationQueryMessage-4.0" schemaLocation="message/ECF-4.0-FeesCalculationQueryMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FeesCalculationResponseMessage-4.0" schemaLocation="message/ECF-4.0-FeesCalculationResponseMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FilingListQueryMessage-4.0" schemaLocation="message/ECF-4.0-FilingListQueryMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FilingListResponseMessage-4.0" schemaLocation="message/ECF-4.0-FilingListResponseMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FilingStatusQueryMessage-4.0" schemaLocation="message/ECF-4.0-FilingStatusQueryMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:FilingStatusResponseMessage-4.0" schemaLocation="message/ECF-4.0-FilingStatusResponseMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageReceiptMessage-4.0" schemaLocation="message/ECF-4.0-MessageReceiptMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:PaymentMessage-4.0" schemaLocation="message/ECF-4.0-PaymentMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:PaymentReceiptMessage-4.0" schemaLocation="message/ECF-4.0-PaymentReceiptMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:RecordDocketingCallbackMessage-4.0" schemaLocation="message/ECF-4.0-RecordDocketingCallbackMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:RecordDocketingMessage-4.0" schemaLocation="message/ECF-4.0-RecordDocketingMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ReviewFilingCallbackMessage-4.0" schemaLocation="message/ECF-4.0-ReviewFilingCallbackMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ServiceInformationQueryMessage-4.0" schemaLocation="message/ECF-4.0-ServiceInformationQueryMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ServiceInformationResponseMessage-4.0" schemaLocation="message/ECF-4.0-ServiceInformationResponseMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:ServiceReceiptMessage-4.0" schemaLocation="message/ECF-4.0-ServiceReceiptMessage.xsd"/>

 <xsd:import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CommonTypes-4.0" schemaLocation="common/ECF-4.0-CommonTypes.xsd"/>

 <xsd:complexType name="GetPolicyRequestType">

 <xsd:sequence>

 <xsd:element ref="policyquery:CourtPolicyQueryMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetPolicy" type="GetPolicyRequestType"/>

 <xsd:complexType name="GetCaseListRequestType">

 <xsd:sequence>

 <xsd:element ref="caselistquery:CaseListQueryMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetCaseList" type="GetCaseListRequestType"/>

 <xsd:complexType name="GetCaseRequestType">

 <xsd:sequence>

 <xsd:element ref="casequery:CaseQueryMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetCase" type="GetCaseRequestType"/>

 <xsd:complexType name="GetDocumentType">

 <xsd:sequence>

 <xsd:element ref="docquery:DocumentQueryMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetDocument" type="GetDocumentType"/>

 <xsd:complexType name="GetFeesCalculationRequestType">

 <xsd:sequence>

 <xsd:element ref="feesquery:FeesCalculationQueryMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetFeesCalculation" type="GetFeesCalculationRequestType"/>

 <xsd:complexType name="GetFilingListRequestType">

 <xsd:sequence>

 <xsd:element ref="listquery:FilingListQueryMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetFilingList" type="GetFilingListRequestType"/>

 <xsd:complexType name="GetFilingStatusRequestType">

 <xsd:sequence>

 <xsd:element ref="statusquery:FilingStatusQueryMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetFilingStatus" type="GetFilingStatusRequestType"/>

 <xsd:complexType name="GetServiceInformationRequestType">

 <xsd:sequence>

 <xsd:element ref="servicequery:ServiceInformationQueryMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetServiceInformation" type="GetServiceInformationRequestType"/>

 <xsd:complexType name="ServeFilingRequestType">

 <xsd:sequence>

 <xsd:element ref="core:CoreFilingMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="ServeFiling" type="ServeFilingRequestType"/>

 <xsd:complexType name="NotifyFilingReviewCompleteRequestType">

 <xsd:annotation>

 <xsd:documentation>Multi-part message type (required for conformance with WS-I Basic

 Profile 1.1</xsd:documentation>

 </xsd:annotation>

 <xsd:sequence>

 <xsd:element ref="reviewcb:ReviewFilingCallbackMessage"/>

 <xsd:element ref="receipt:PaymentReceiptMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="NotifyFilingReviewCompleteRequest" type="NotifyFilingReviewCompleteRequestType"/>

 <xsd:complexType name="NotifyFilingReviewCompleteType">

 <xsd:sequence>

 <xsd:element ref="NotifyFilingReviewCompleteRequest"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="NotifyFilingReviewComplete" type="NotifyFilingReviewCompleteType"/>

 <xsd:complexType name="RecordFilingRequestType">

 <xsd:annotation>

 <xsd:documentation>Multi-part message type (required for conformance with WS-I Basic

 Profile 1.1</xsd:documentation>

 </xsd:annotation>

 <xsd:sequence>

 <xsd:element ref="docket:RecordDocketingMessage" maxOccurs="unbounded"/>

 <xsd:element ref="core:CoreFilingMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="RecordFilingRequest" type="RecordFilingRequestType"/>

 <xsd:complexType name="RecordFilingType">

 <xsd:sequence>

 <xsd:element ref="RecordFilingRequest"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="RecordFiling" type="RecordFilingType"/>

 <xsd:complexType name="ReviewFilingRequestType">

 <xsd:annotation>

 <xsd:documentation>Multi-part message type (required for conformance with WS-I Basic

 Profile 1.1</xsd:documentation>

 </xsd:annotation>

 <xsd:sequence>

 <xsd:element ref="core:CoreFilingMessage"/>

 <xsd:element ref="payment:PaymentMessage" minOccurs="0" maxOccurs="1"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="ReviewFilingRequest" type="ReviewFilingRequestType"/>

 <xsd:complexType name="ReviewFilingType">

 <xsd:sequence>

 <xsd:element ref="ReviewFilingRequest"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="ReviewFiling" type="ReviewFilingType"/>

 <xsd:complexType name="NotifyDocketingCompleteRequestType">

 <xsd:sequence>

 <xsd:element ref="docketcb:RecordDocketingCallbackMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="NotifyDocketingComplete" type="NotifyDocketingCompleteRequestType"/>

 <xsd:complexType name="GetPolicyResponseType">

 <xsd:sequence>

 <xsd:element ref="policyresponse:CourtPolicyResponseMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetPolicyResponse" type="GetPolicyResponseType"/>

 <xsd:complexType name="GetCaseListResponseType">

 <xsd:sequence>

 <xsd:element ref="caselistresponse:CaseListResponseMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetCaseListResponse" type="GetCaseListResponseType"/>

 <xsd:complexType name="GetCaseResponseType">

 <xsd:sequence>

 <xsd:element ref="caseresponse:CaseResponseMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetCaseResponse" type="GetCaseResponseType"/>

 <xsd:complexType name="GetDocumentResponseType">

 <xsd:sequence>

 <xsd:element ref="docresponse:DocumentResponseMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetDocumentResponse" type="GetDocumentResponseType"/>

 <xsd:complexType name="GetFeesCalculationResponseType">

 <xsd:sequence>

 <xsd:element ref="feesresponse:FeesCalculationResponseMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetFeesCalculationResponse" type="GetFeesCalculationResponseType"/>

 <xsd:complexType name="GetFilingListResponseType">

 <xsd:sequence>

 <xsd:element ref="listresponse:FilingListResponseMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetFilingListResponse" type="GetFilingListResponseType"/>

 <xsd:complexType name="GetFilingStatusResponseType">

 <xsd:sequence>

 <xsd:element ref="statusresponse:FilingStatusResponseMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetFilingStatusResponse" type="GetFilingStatusResponseType"/>

 <xsd:complexType name="GetServiceInformationResponseType">

 <xsd:sequence>

 <xsd:element ref="serviceresponse:ServiceInformationResponseMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="GetServiceInformationResponse" type="GetServiceInformationResponseType"/>

 <xsd:complexType name="ServeFilingResponseType">

 <xsd:sequence>

 <xsd:element ref="servicereceipt:ServiceReceiptMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="ServeFilingResponse" type="ServeFilingResponseType"/>

 <xsd:complexType name="NotifyFilingReviewCompleteResponseType">

 <xsd:sequence>

 <xsd:element ref="message:MessageReceiptMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="NotifyFilingReviewCompleteResponse" type="NotifyFilingReviewCompleteResponseType"/>

 <xsd:complexType name="RecordFilingResponseType">

 <xsd:sequence>

 <xsd:element ref="message:MessageReceiptMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="RecordFilingResponse" type="RecordFilingResponseType"/>

 <xsd:complexType name="ReviewFilingResponseType">

 <xsd:sequence>

 <xsd:element ref="message:MessageReceiptMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="ReviewFilingResponse" type="ReviewFilingResponseType"/>

 <xsd:complexType name="NotifyDocketingCompleteResponseType">

 <xsd:sequence>

 <xsd:element ref="message:MessageReceiptMessage"/>

 </xsd:sequence>

 </xsd:complexType>

 <xsd:element name="NotifyDocketingCompleteResponse" type="NotifyDocketingCompleteResponseType"/>

</xsd:schema>

Example 2 - ECF 4.01 adapted CourtRecordMDE.wsdl

<?xml version="1.0" encoding="UTF-8"?>

<!--

 LegalXML Electronic Court Filing Version 4.1

 Copyright (c) OASIS Open 2022. All Rights Reserved.

 Source: http://docs.oasis-open.org/legalxml-courtfiling/ecf/v4.1

-->

<!--

 Modified for use with ECF 4.01

-->

<!--

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

-->

<definitions targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0"

 xmlns:tns="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0"

 xmlns:wrappers="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"

 xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"

 xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200702"

 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-361 wssecurity-utility-1.0.xsd">

 <wsp:UsingPolicy wsdl:required="true"/>

 <wsp:Policy wsu:Id="MyPolicy">

 <wsrmp:RMAssertion/>

 </wsp:Policy>

 <wsdl:types>

 <xsd:schema>

 <xsd:import

 namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0"

 schemaLocation="../xsd/wrappers.xsd"/>

 </xsd:schema>

 </wsdl:types>

 <!-- messages -->

 <message name="GetCaseListRequest">

 <part name="CaseListQueryMessage" element="wrappers:GetCaseList"/>

 </message>

 <message name="GetCaseListResponse">

 <part name="CaseListResponseMessage" element="wrappers:GetCaseListResponse"/>

 </message>

 <message name="GetCaseRequest">

 <part name="CaseQueryMessage" element="wrappers:GetCase"/>

 </message>

 <message name="GetCaseResponse">

 <part name="CaseResponseMessage" element="wrappers:GetCaseResponse"/>

 </message>

 <message name="GetDocumentRequest">

 <part name="DocumentQueryMessage" element="wrappers:GetDocument"/>

 </message>

 <message name="GetDocumentResponse">

 <part name="DocumentResponseMessage" element="wrappers:GetDocumentResponse"/>

 </message>

 <message name="GetServiceInformationRequest">

 <part name="ServiceInformationQueryMessage" element="wrappers:GetServiceInformation"/>

 </message>

 <message name="GetServiceInformationResponse">

 <part name="ServiceInformationResponseMessage"

 element="wrappers:GetServiceInformationResponse"/>

 </message>

 <message name="RecordFilingRequest">

 <part name="RecordFilingRequestMessage" element="wrappers:RecordFiling"/>

 </message>

 <message name="RecordFilingResponse">

 <part name="MessageReceiptMessage" element="wrappers:RecordFilingResponse"/>

 </message>

 <!-- ports -->

 <portType name="CourtRecordMDE">

 <operation name="GetCase">

 <input message="tns:GetCaseRequest"/>

 <output message="tns:GetCaseResponse"/>

 </operation>

 <operation name="GetCaseList">

 <input message="tns:GetCaseListRequest"/>

 <output message="tns:GetCaseListResponse"/>

 </operation>

 <operation name="GetDocument">

 <input message="tns:GetDocumentRequest"/>

 <output message="tns:GetDocumentResponse"/>

 </operation>

 <operation name="GetServiceInformation">

 <input message="tns:GetServiceInformationRequest"/>

 <output message="tns:GetServiceInformationResponse"

 />

 </operation>

 <operation name="RecordFiling">

 <input message="tns:RecordFilingRequest"/>

 <output message="tns:RecordFilingResponse"/>

 </operation>

 </portType>

 <!-- bindings -->

 <binding name="CourtRecordMDESoap" type="tns:CourtRecordMDE">

 <wsp:PolicyReference URI="#MyPolicy"/>

 <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>

 <operation name="GetCase">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0\GetCase"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="GetCaseList">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0\GetCaseList"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="GetDocument">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0\GetDocument"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="GetServiceInformation">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0\GetServiceInformation"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="RecordFiling">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0\RecordFiling"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 </binding>

 <!-- services not defined here...defined in an implementation-specific WSDL that imports this one -->

</definitions>

Example 3 - ECF 4.01 adapted FilingAssemblyMDE.wsdl

<?xml version="1.0" encoding="UTF-8"?>

<!--

 LegalXML Electronic Court Filing Version 4.1

 Copyright (c) OASIS Open 2022. All Rights Reserved.

 Source: http://docs.oasis-open.org/legalxml-courtfiling/ecf/v4.1

-->

<!--

 Modified for use with ECF 4.01

-->

<!--

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

-->

<definitions targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0"

 xmlns:tns="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0"

 xmlns:wrappers="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"

 xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"

 xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200702"

 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-361 wssecurity-utility-1.0.xsd">

 <wsp:UsingPolicy wsdl:required="true"/>

 <wsp:Policy wsu:Id="MyPolicy">

 <wsrmp:RMAssertion/>

 </wsp:Policy>

 <wsdl:types>

 <xsd:schema>

 <xsd:import

 namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0"

 schemaLocation="../xsd/wrappers.xsd"/>

 </xsd:schema>

 </wsdl:types>

 <!-- messages -->

 <message name="NotifyFilingReviewCompleteRequest">

 <part name="NotifyFilingReviewCompleteRequestMessage"

 element="wrappers:NotifyFilingReviewComplete"/>

 </message>

 <message name="NotifyFilingReviewCompleteResponse">

 <part name="MessageReceiptMessage" element="wrappers:NotifyFilingReviewCompleteResponse"/>

 </message>

 <!-- ports -->

 <portType name="FilingAssemblyMDE">

 <operation name="NotifyFilingReviewComplete">

 <input message="tns:NotifyFilingReviewCompleteRequest"/>

 <output message="tns:NotifyFilingReviewCompleteResponse"/>

 </operation>

 </portType>

 <!-- bindings -->

 <binding name="FilingAssemblyMDESoap" type="tns:FilingAssemblyMDE">

 <wsp:PolicyReference URI="#MyPolicy"/>

 <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>

 <operation name="NotifyFilingReviewComplete">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfilingschema:wsdl:FilingAssemblyMDE-4.0\NotifyFilingReviewComplete"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 </binding>

 <!-- services not defined here...defined in an implementation-specific WSDL that imports this one -->

</definitions>

Example 3 - ECF 4.01 adapted FilingReviewMDE.wsdl

<?xml version="1.0" encoding="UTF-8"?>

<!--

 LegalXML Electronic Court Filing Version 4.1

 Copyright (c) OASIS Open 2022. All Rights Reserved.

 Source: http://docs.oasis-open.org/legalxml-courtfiling/ecf/v4.1

-->

<!--

 Modified for use with ECF 4.01

-->

<!--

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

-->

<definitions targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingReviewMDE-4.0"

 xmlns:tns="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingReviewMDE-4.0"

 xmlns:wrappers="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"

 xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"

 xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200702"

 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-361 wssecurity-utility-1.0.xsd">

 <wsp:UsingPolicy wsdl:required="true"/>

 <wsp:Policy wsu:Id="MyPolicy">

 <wsrmp:RMAssertion/>

 </wsp:Policy>

 <wsdl:types>

 <xsd:schema>

 <xsd:import

 namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0"

 schemaLocation="../xsd/wrappers.xsd"/>

 </xsd:schema>

 </wsdl:types>

 <!-- messages -->

 <message name="GetFeesCalculationRequest">

 <part name="FeesCalculationQueryMessage" element="wrappers:GetFeesCalculation"/>

 </message>

 <message name="GetFeesCalculationResponse">

 <part name="FeesCalculationResponseMessage" element="wrappers:GetFeesCalculationResponse"/>

 </message>

 <message name="GetFilingListRequest">

 <part name="FilingListQueryMessage" element="wrappers:GetFilingList"/>

 </message>

 <message name="GetFilingListResponse">

 <part name="FilingListResponseMessage" element="wrappers:GetFilingListResponse"/>

 </message>

 <message name="GetFilingStatusRequest">

 <part name="FilingStatusQueryMessage" element="wrappers:GetFilingStatus"/>

 </message>

 <message name="GetFilingStatusResponse">

 <part name="FilingStatusReponseMessage" element="wrappers:GetFilingStatusResponse"/>

 </message>

 <message name="GetPolicyRequest">

 <part name="CourtPolicyQueryMessage" element="wrappers:GetPolicy"/>

 </message>

 <message name="GetPolicyResponse">

 <part name="CourtPolicyResponseMessage" element="wrappers:GetPolicyResponse"/>

 </message>

 <message name="NotifyDocketingCompleteRequest">

 <part name="RecordDocketingMessage" element="wrappers:NotifyDocketingComplete"/>

 </message>

 <message name="NotifyDocketingCompleteResponse">

 <part name="MessageReceiptMessage" element="wrappers:NotifyDocketingCompleteResponse"/>

 </message>

 <message name="ReviewFilingRequest">

 <part name="ReviewFilingRequestMessage" element="wrappers:ReviewFiling"/>

 </message>

 <message name="ReviewFilingResponse">

 <part name="MessageReceiptMessage" element="wrappers:ReviewFilingResponse"/>

 </message>

 <!-- ports -->

 <portType name="FilingReviewMDE">

 <operation name="GetFeesCalculation">

 <input message="tns:GetFeesCalculationRequest"/>

 <output message="tns:GetFeesCalculationResponse"/>

 </operation>

 <operation name="GetFilingList">

 <input message="tns:GetFilingListRequest"/>

 <output message="tns:GetFilingListResponse"/>

 </operation>

 <operation name="GetFilingStatus">

 <input message="tns:GetFilingStatusRequest"/>

 <output message="tns:GetFilingStatusResponse"/>

 </operation>

 <operation name="GetPolicy">

 <input message="tns:GetPolicyRequest"/>

 <output message="tns:GetPolicyResponse"/>

 </operation>

 <operation name="NotifyDocketingComplete">

 <input message="tns:NotifyDocketingCompleteRequest"/>

 <output message="tns:NotifyDocketingCompleteResponse"/>

 </operation>

 <operation name="ReviewFiling">

 <input message="tns:ReviewFilingRequest"/>

 <output message="tns:ReviewFilingResponse"/>

 </operation>

 </portType>

 <!-- bindings -->

 <binding name="FilingReviewMDESoap" type="tns:FilingReviewMDE">

 <wsp:PolicyReference URI="#MyPolicy"/>

 <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>

 <operation name="GetFeesCalculation">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0\GetFeesCalculation"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="GetFilingList">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0\GetFilingList"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="GetFilingStatus">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0\GetFilingStatus"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="GetPolicy">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0\GetPolicy"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="NotifyDocketingComplete">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0\NotifyDocketingComplete"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 <operation name="ReviewFiling">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0\ReviewFiling"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 </binding>

 <!-- services not defined here...defined in an implementation-specific WSDL that imports this one -->

</definitions>

Example 3 - ECF 4.01 adapted ServiceMDE.wsdl

<?xml version="1.0" encoding="UTF-8"?>

<!--

 LegalXML Electronic Court Filing Version 4.1

 Copyright (c) OASIS Open 2022. All Rights Reserved.

 Source: http://docs.oasis-open.org/legalxml-courtfiling/ecf/v4.1

-->

<!--

 Modified for use with ECF 4.01

-->

<!--

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

-->

<definitions targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:ServiceMDE-4.0"

 xmlns:tns="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:ServiceMDE-4.0"

 xmlns:wrappers="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"

 xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"

 xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200702"

 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-361 wssecurity-utility-1.0.xsd">

 <wsp:UsingPolicy wsdl:required="true"/>

 <wsp:Policy wsu:Id="MyPolicy">

 <wsrmp:RMAssertion/>

 </wsp:Policy>

 <wsdl:types>

 <xsd:schema>

 <xsd:import

 namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.0"

 schemaLocation="../xsd/wrappers.xsd"/>

 </xsd:schema>

 </wsdl:types>

 <!-- messages -->

 <message name="ServeFilingRequest">

 <part name="CoreFilingMessage" element="wrappers:ServeFiling"/>

 </message>

 <message name="ServeFilingResponse">

 <part name="ServiceReceiptMessage" element="wrappers:ServeFilingResponse"/>

 </message>

 <!-- ports -->

 <portType name="ServiceMDE">

 <operation name="ServeFiling">

 <input message="tns:ServeFilingRequest"/>

 <output message="tns:ServeFilingResponse"/>

 </operation>

 </portType>

 <!-- bindings -->

 <binding name="ServiceMDESoap" type="tns:ServiceMDE">

 <wsp:PolicyReference URI="#MyPolicy"/>

 <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>

 <operation name="ServeFiling">

 <soap:operation

 soapAction="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:ServiceMDE-4.0\ServeFiling"/>

 <input>

 <soap:body use="literal"/>

 </input>

 <output>

 <soap:body use="literal"/>

 </output>

 </operation>

 </binding>

 <!-- services not defined here...defined in an implementation-specific WSDL that imports this one -->

</definitions>

Example 3 - ECF 4.01 adapted CourtRecordMDE-ImplementationExample.wsdl

<?xml version="1.0" encoding="UTF-8"?>

<!--

This WSDL document is a sample of how to import the main OASIS LegalXML Electronic Court Filing v4.1 Web Services Profile Definitions to define implementing physical services for one or more Major Design Elements.

-->

<!--

 Modified for use with ECF 4.01

-->

<definitions

 targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:WebServices-ImplementationExample-4.0"

 xmlns:wsmp="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema"

 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

 xmlns="http://schemas.xmlsoap.org/wsdl/">

 <import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.0" location="../CourtRecordMDE.wsdl"/>

 <service name="CourtRecordMDEService">

 <port name="CourtRecordMDE" binding="wsmp:CourtRecordMDESoap">

 <soap:address location="https://localhost/..."/>

 </port>

 </service>

</definitions>

Example 3 - ECF 4.01 adapted FilingAssemblyMDE-ImplementationExample.wsdl

<?xml version="1.0" encoding="UTF-8"?>

<!--

This WSDL document is a sample of how to import the main OASIS LegalXML Electronic Court Filing v4.1 Web Services Profile Definitions to define implementing physical services for one or more Major Design Elements.

-->

<!--

 Modified for use with ECF 4.01

-->

<definitions

 targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:WebServices-ImplementationExample-4.0"

 xmlns:wsmp="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema"

 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

 xmlns="http://schemas.xmlsoap.org/wsdl/">

 <import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingAssemblyMDE-4.0" location="../FilingAssemblyMDE.wsdl"/>

 <service name="FilingAssemblyMDEService">

 <port name="FilingAssemblyMDE" binding="wsmp:FilingAssemblyMDESoap">

 <soap:address location="https://localhost/..."/>

 </port>

 </service>

</definitions>

Example 3 - ECF 4.01 adapted FilingReviewMDE-ImplementationExample.wsdl

<?xml version="1.0" encoding="UTF-8"?>

<!--

This WSDL document is a sample of how to import the main OASIS LegalXML Electronic Court Filing v4.1 Web Services Profile Definitions to define implementing physical services for one or more Major Design Elements.

-->

<!--

 Modified for use with ECF 4.01

-->

<definitions

 targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:WebServices-ImplementationExample-4.0"

 xmlns:wsmp="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingReviewMDE-4.0"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema"

 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

 xmlns="http://schemas.xmlsoap.org/wsdl/">

 <import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:FilingReviewMDE-4.0" location="../FilingReviewMDE.wsdl"/>

 <service name="FilingReviewMDEService">

 <port name="FilingReviewMDE" binding="wsmp:FilingReviewMDESoap">

 <soap:address location="https://localhost/..."/>

 </port>

 </service>

</definitions>

Example 3 - ECF 4.01 adapted ServiceMDE-ImplementationExample.wsdl

<?xml version="1.0" encoding="UTF-8"?>

<!--

This WSDL document is a sample of how to import the main OASIS LegalXML Electronic Court Filing v4.1 Web Services Profile Definitions to define implementing physical services for one or more Major Design Elements.

-->

<!--

 Modified for use with ECF 4.01

-->

<definitions

 targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:WebServices-ImplementationExample-4.0"

 xmlns:wsmp="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:ServiceMDE-4.0"

 xmlns:xsd="http://www.w3.org/2001/XMLSchema"

 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

 xmlns="http://schemas.xmlsoap.org/wsdl/">

 <import namespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:ServiceMDE-4.0" location="../ServiceMDE.wsdl"/>

 <service name="ServiceMDEService">

 <port name="ServiceMDE" binding="wsmp:ServiceMDESoap">

 <soap:address location="https://localhost/..."/>

 </port>

 </service>

</definitions>

1. Revision History

[Optional section]

Revisions made since the initial stage of this numbered Version of this document are tracked here.

Note: If revision tracking is handled in another system like github, provide a link to it instead of using this table, if desired. Remove this note before submitting for publication.

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Editor** | **Changes Made** |
| 0.1 | 08-17-2023 | Gary Graham | Initial draft. |

1. Core Specification Normative Requirements

The following table provides a listing of all normative statements from the Core ECF v4.1 specification:

|  |  |  |
| --- | --- | --- |
| **No.** | **Section/Line #** | **Requirement** |
| 1 | 2.1 | In order to be compliant, an implementation of the ECF specification MUST implement the core specification and at least one service interaction profile and one document signature profile. |
| 2 | 2.2 | An ECF 4.1-compliant implementation may implement one or more of the MDEs defined in the specification but a complete ECF 4.1 system MUST include at least one each of the Filing Assembly, Filing Review and Court Record MDEs. |
| 3 | 2.2 | When multiple MDEs are implemented by a single court, vendor or application, the application MUST maintain the ECF 4.1 specified operations between each MDE so that other applications will be able to interoperate with it. |
| 4 ? | 2.2 | In order to be compliant with ECF 4.1, an MDE MUST support all messages required for that MDE. |
| 5 & 6 | 2.4 | The court MUST have only one active, authoritative version of its policies at a given time; both the human-readable and the machine-readable statements of those policies MUST have the same release dates for the court.  |
| 7 | 2.4 | The court’s human-readable and machine-readable court policies MUST each have a version numbering method associated with it.  |
| 8 | 2.4.1 | To be compliant with the ECF 4.1 specification, each court MUST publish a human-readable court policy |
| 9 | 2.4.1 | human-readable court policy MUST include each of the following:1. The unique court identifier2. The location of the machine-readable court policy3. A definition of what constitutes a “lead document” in the court4. A description of how filer identifiers are to be maintained during electronic communications regarding the case5. A description of how the court processes (dockets) filings6. A description of any instances in which the court will mandate an element that the ECF 4.1 schema makes optional7. A description of any restrictions to data property values other than code list restrictions. (This restriction may be removed in later versions of the ECF specification)8. Any other rules required for electronic filing in the court |
| 10 | 2.4.2 | The machine-readable court policy MUST be provided to the Filing Assembly MDE either by the Filing Review MDE through the GetCourtPolicy query or some other means. |
| 11 | 2.4.5 | If court-specific constraint schemas are used, instance documents MUST validate against both the ECF schemas and the court constraint schemas. |
| 12 | 3.1 | The operations in bold are required and MUST occur in every successful filing as long as sending and receiving MDEs are implemented in separate systems.  |
| 13 | 3.2 | Successful queries MUST return an <ecf:ErrorCode> of “0”.  |
| 14 | 3.2 | Failed queries MUST NOT return an <ecf:ErrorCode> of “0” |
| 15 | 3.2.2 | the Court Record MDE MUST have access to the court’s registry with all updated information about case participants. |
| 16 | 3.2.2 | There MUST be only one such registry per court, |
| 17 | 3.2.2 | If the court provides a Hub Service MDE, the electronic service information returned from this query MUST include the court’s Service MDE ID for all case participants who have one. |
| 18. | 3.2.4 | The Filing Assembly MDE MUST submit the filing to the court by invoking the ReviewFiling operation on the Filing Review MDE. |
| 19.  | 3.2.5 | This operation (i.e., ServeFiling) MUST NOT be used to serve parties in a new case or to persons or organizations that have not yet been made party to the case.  |
| 20. | 3.2.5 | The hub Service MDE MUST then broadcast the message to each of the individual Legal Service MDE’s ServeFiling operations and respond synchronously with a single ServiceResponseMessage to the Filing Assembly MDE |
| 21. | 3.2.5 | If a court chooses to support electronic service, then each Filing Assembly MDE MUST support service operations for the clients for which it provides Filing Assembly functionality. |
| 22. | 3.2.6 | If the clerk reviews and accepts the filing, the Filing Review MDE MUST invoke the RecordFiling operation on the Court Record MDE.  |
| 23. | 3.2.7 | If the <RequireAsynchronousResponsesIndicator> in the court policy is “true”, the Court Record MDE MUST invoke the NotifyDocketingComplete operation on the Filing Review MDE as a callback message to the RecordFiling operation |
| 24. | 3.2.7 | If the Court Record MDE rejected the filing, an explanation MUST be provided. |
| 25.  | 3.2.7 | If the Court Record MDE accepts the filing, the docketing information (e.g. date and time the document was entered into the court record, judge assigned, document identifiers and next court event scheduled) MUST be provided. |
| 26. | 3.2.8 | If the clerk rejects the filings or the Filing Review MDE receives the NotifyDocketingComplete message and the <RequireAsynchronousResponsesIndicator> in the court policy is “true”, the Filing Review MDE MUST invoke the NotifyFilingReviewComplete operation on the Filing Assembly MDE |
| 27. | 3.2.8 | The operation MAY return the filed documents or links to the documents, but MUST include the [FIPS 180-4] SHA 256 document hash, |
| 28. | 3.2.8 | If the filing included a payment, and the filing was accepted by the clerk and court record system, a receipt for the payment MUST be included in the operation.  |
| 29. | 3.3.1.1 | Attachment identifiers MUST be unique within a message transmission. |
| 30. | 3.3.1.2 | Case identifiers (case numbers) are assigned by the court record system and MUST be unique within a court. |
| 31.  | 3.3.1.3 | Court identifiers are locally assigned by the court administrator for a region (typically a state, provincial or federal court administrator) and MUST be universally unique to a court but not necessarily to a particular court house, branch or subunit of a court. |
| 32. | 3.3.1.3 | Court identifiers MUST conform to following convention: <Internet domain of the court administrator>:<unique identifier within the court system>.  |
| 33. | 3.3.1.4 | Document identifiers are assigned by the court record system and MUST be unique within a court. |
| 34. | 3.3.1.5 | Filing identifiers MUST be unique within a court and will be generated by the court in response to a ReviewFiling operation. |
| 35. | 3.3.1.6 | The address of an MDE MUST be unique within a given communications infrastructure. |
| 36. | 3.3.1.7 | If the <RequireAsynchronousResponsesIndicator> in the CourtPolicyResponseMessage is “true”, then both <SendingMDELocationID> and <SendingMDEProfileCode> MUST be included in all ECF 4.1 messages that include these elements. |
| 37. | 3.3.1.8 | Identifiers for filers and parties to a case, both persons and organizations, MUST be unique within a case and will be generated by the court in response to a ReviewFiling operation.  |
| 38. | 3.3.3.1 | A CoreFilingMessage MUST express the name or names of the party or parties on whose behalf a document is filed, and the party whose document is the subject of a responsive document being submitted for filing. |
| 39. | 3.3.3.1 | If a CoreFilingMessage includes documents, the message MUST include only one level of connected and supporting documents. |
| 40. | 3.4 | All ROA (Record on Appeal) transactions, either the original filing or subsequent amendments, MUST contain, as the lead document, an Index of Record document that itemizes the content of the record on appeal. |
| 41. | 3.4 | All ROA documents being submitted, including the Index of Record document and each document within the record, MUST have at least one court-defined document type that indicates the type of transaction to be performed on the document, and whether the document is being added to or stricken from the record.  |
| 42. | 3.4 | When a document within the ROA transaction is being stricken from the court record, the document MUST be identified by the unique document identifier, which was provided by the Court Record MDE when the document was initially filed (See section 3.3.1.4). |
| 43. | 3.4 | A hierarchical structure of case lineage elements MUST be used to express the target case’s predecessor cases at prior courts. Each predecessor case MAY also have its own predecessor case, as necessary to express the full lineage of an appellate case. |
| 44. | 3.4 | When the ROA transaction is electronically transferred from one court to another, the target case number in the destination court and the case lineage, which includes the predecessor case number in the sending court, MUST be provided. |
| 45.46. | 3.4 | If the ROA transaction is a case initiating filing in the destination court, then the <FilingCase> object MUST be present and the <CaseTrackingID> MUST be absent. |
| 47. | 3.4 | If the ROA transaction is a case initiating filing in the destination court, then the <FilingCase> object MUST be present and the <CaseTrackingID> MUST be absent. |
| 48. | 3.4 | When a ROA amendment transaction is sent, the Index of Record document MUST reflect the status of the record assuming that the transaction will be accepted. |
| 49. | 3.4 | Individual documents within the ROA transaction MUST not be individually accepted or rejected. |
| 50. | 3.4 | All documents within the ROA transaction MUST have the same acceptance or rejection disposition. |
|  |  |  |

1. Example Title

Additional Appendixes may be added as needed:

* IANA Considerations
	1. Subsidiary section

text

* + 1. Sub-subsidiary section

Text

* + - 1. Sub-sub-subsidiary section

text

* + - * 1. Sub-sub-sub-subsidiary section

text

1. Notices

Copyright © OASIS Open 2023. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](https://www.oasis-open.org/policies-guidelines/ipr/) may be found at the OASIS website: [<https://www.oasis-open.org/policies-guidelines/ipr/>].

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. OASIS AND ITS MEMBERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THIS DOCUMENT OR ANY PART THEREOF.

The name "OASIS" is a trademark of [OASIS](https://www.oasis-open.org/), the owner and developer of this document, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, documents, while reserving the right to enforce its marks against misleading uses. Please see <https://www.oasis-open.org/policies-guidelines/trademark/> for above guidance.