**Electronic Court Filing v4.1 Public Review Feedback**

This document provides public review Feedback to the Electronic Court Filing (ECF) v4.1 portion of the first Public Review announced Feb. 2, 2023.

1. The xsd folder includes wrappers.xsd, wrappers-2.xsd, and wrappers-old.xsd.

Recommendation: Only wrappers.xsd should be provided, wrappers-2.xsd and wrappers-old.xsd should be removed.

2. In wrappers.xsd, clean-up file header comments.

Recommendation: Lines 2 to 62 should be deleted.

3. In Related Work, as “replaced or superseded”, ECF v4.01 Errata 01 appears to be missing.

Also note that the ECF v4.01 Errata02 specification does not specify ECF v4.01 Errata 01 as replaced or superseded, so there is no chain of replacement/supersession that includes ECF v4.01 Errata 01.

Recommendation: Include ECF v4.01 Errata 01 as replaced or superseded.

4. Section 1.2 ‘Relationship to Prior Specifications’ addresses backward compatibility and interoperability.

Although this section specifies that “ECF 4.1 s not backward-compatible with ECF 3.0, 3.01 and 3.1”, it says nothing about backward compatibility/interoperability between ECF 4.0, ECF 4.01 and ECF 4.1. The implication is that ECF 4.0, ECF 4.01, and ECF 4.1 are backward-compatible and can interoperate.

Is ECF 4.1 backwards compatible with ECF 4.01? Is ECF 4.1 interoperable with ECF 4.01?

What does it mean to be backwards compatible? Does this mean that ‘messages’ created under ECF 4.01 can be used with ECF 4.1?

If ‘message’ means an XML instance conformant with one of the schemas in the xsd/message folder, then ECF 4.1 messages should be backward compatible with ECF 4.01 since (by my recollection) no new elements have been included, and no constraints have been tightened, e.g., some maxOccurs attribute values have been ‘relaxed’ which is loosening, not tightening).

However, if ‘message’ implies ‘an ‘exchange’ (such as a Request or Response and not an ECF 4.1 message (e.g., a Request may contain one or more ECF messages)) then ECF v4.1 may not be backwardly compatible (at least when using Web Services SIP) since ECF 4.1 exchanges, using [ECF-WS-SIP-v4.1], now must be ‘wrapped’.

Furthermore, if ‘backward compatible’ means that a fully compliant ECF 4.01 MDE can seamlessly exchange requests and responses with another fully compliant ECF 4.1 (and vice versa) then the answer is likely no (or is this interoperability?).

Note: interoperability between ECF 4.01 and ECF 4.1 may be possible if the ECF 4.1 wrappers.xsd and the ECF 4.1 WS-SIP is used with ECF 4.01. The TC should consider writing a ‘non-standards’ track document that provides guidance.

Recommendation: Make it clear that ECF v4.1 is or is not backward compatible with ECF v4.0 and ECF v4.01.

5. Message Stream

The illustration in section 2.3.3 Sample Message Streams (i.e., Figure 1. Simple Message Stream) does not show the containment provided by the new wrappers.xsd.

Is the use of wrappers.xsd optional?

There are no statements within the prose specification requiring the use of wrappers.xsd. Furthermore, there are no ECF v4.1 provided schemas that require the application of wrappers.xsd. The wrappers.xsd is required for use with the ECF v4.1 Web Services SIP.

It appears that wrappers.xsd is an included, but optional, implementation allowable accessory.

Recommendation: Determine the role for wrappers.xsd in ECF v4.1 (also consider #7 below). Upon role determination, revise the prose specification (and possibly other technical artifacts) as appropriate, such as making it clear whether it is required or optional. This may include revisions in Appendix C.

6. Section 4.2 ‘ECF 4.1 Common Schemas’ does not include wrappers.xsd.

The new wrappers.xsd should be included in the schema section, but in which subsection should it appear? None of the existing subsections (i.e., Case Type Schemas, Common Schemas, Constraint and Subset Schemas, or Message Schemas) appear correct. Should a new subsection be added?

Recommendation: Include wrappers.xsd in the schema section, within a new ‘Exchange Schema’ subsection.

7. Does wrappers.xsd really belong in ECF 4.1 Web Services SIP?

Note that the 2022-06-18 entry in Revision History (Appendix G) provides: “Added xsd/wrappers.xsd to support document/literal web services.” Note that “document/literal” refers to the Web Services SIP.

Also observe that wrappers.xsd is not employed in the Portable Media SIP or the IBM MQ SIP. As such it is uniquely associated with, and perhaps should be a part of, the Web Services SIP.

Also consider that if wrappers.xsd is an artifact of [ECF-WS-SIP-v4.1] then this would likely resolve the apparent backward compatibility concern.

Recommendation:

a. Presently, neither Portable Media SIP nor IBM MQ SIP require the use of wrappers.xsd. The wrappers.xsd exists exclusively for Web Services SIP. As such, warppers.xsd should be an artifact of Web Services SIP and not an artifact of ECF v4.1. Remove wrappers.xsd from ECF v4.1 and include wrappers.xsd within ECF v4.1 Web Services SIP.

b. In the alternative, under which wrappers.xsd remains an artifact of ECF v4.1, then revise both the Portable Media SIP and the IBM MQ SIP to incorporate wrappers.xsd.

8. Example ECF 4.1 messages (in ecf-v4.1-csd01/xml folder) provide confusing header comments. Two separate comments are included in most of the examples, and each comment identifies a different ECF version as shown below:

<!--

Electronic Court Filing Web Services Service Interaction Profile Version 4.1

Committee Specification Draft 01

07 December 2022

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

Source: https://docs.oasis-open.org/legalxml-courtfiling/ecf-webservices/v4.1/csd01/xsd/

Latest version of narrative specification: https://docs.oasis-open.org/legalxml-courtfiling/ecf-webservices/v4.1/ecf-webservices-v4.1.html

TC IPR Statement: https://www.oasis-open.org/committees/legalxml-courtfiling/ipr.php

-->

<!--

Electronic Court Filing Version 4.01 Plus Errata 02

OASIS Standard incorporating Approved Errata 02

07 July 2015

Copyright (c) OASIS Open 2015. All Rights Reserved.

Source: http://docs.oasis-open.org/legalxml-courtfiling/specs/ecf/v4.01/ecf-v4.01-spec/errata02/os/xsd/

-->

Since namespace URIs have been updated for v4.1, these examples are no longer valid for v4.01.

See ECF-4.1-CaseListResponseMessage.xml, ECF-4.1-CaseQueryMessage.xml, ECF-4.1-CaseResponseMessage.xml, ECF-4.1-CoreFilingMessage-Criinal.xml, ECF-4.1-CourtPolicyQueryMessage.xml, ECF-4.1-CourtPolicyResponseMessage.xml, ECF-4.1-DocumentQueryMessage.xml, ECF-41.-DocumentResponseMessage.xml, ECF-4.1-FeesCalculationResponseMessage.xml, ECF-4.1-FilingListQueryMessage.xml, ECF-41.-FilingListResponseMessage.xml, ECF-4.1-FilingStatusQueryMessage.xml, ECF-4.1-FilingStatusResponseMessage.xml, ECF-4.1-MessageReceiptMessage.xml, ECF-4.1-PaymentMessage.xml, ECF-4.1-PaymentReceiptMesage.xml, ECF-4.1-RecordDocketingMessage.xml, ECF-4.1-ServiceInformationQueryMessage.xml, ECF-4.1-ServiceInformationResponseMessage.xml, & ECF-4.1-ServiceReceiptMessage.xml.

Recommendation: Ensure all ECF 4.1 message examples provide a single version relevant comments header.

9. Portable Media SIP

Section 5.3 ‘Supported Service Interaction Profiles’ includes the Portable Media SIP 1.01. Should this reference ‘Electronic Court Filing 4.0 Portable Media Service Interaction Profile Version 2.0’ (Committee Draft 01, 21- September 2008) instead?

Note that ‘Portable Media Messaging Profile 1.0’ (Committee Draft, November 15, 2005) is intended for use with ECF 3.0.

Portable Media SIP v2.0 specifies (Introduction): “This document is a Proposed Standard developed by the OASIS LegalXML Member Section’s Electronic Court Filing (ECF) Technical Committee that defines a service interaction profile for use with the ECF 4.0 specification that does not require an active network connection.”

Does a new version of the Portable Media specification, that specifically includes [ECF-v4.1], need to be produced or is it sufficient for the [ECF-v4.1] specification to prescribe Portable Media SIP compatibility in its section 5.3?

If considering an updated Portable Media SIP (PM-SIP) specification for ECF 4.1, then consider that the PM-SIP inadequately addresses exchanges other than the Review Filing Request exchange between an FAMDE and an FRMDE. The PM-SIP cannot be satisfactorily used for any other exchange. To wit, the PM-SIP, section 2.4 ‘Operation Addressing’ states: “In this version of the service interaction profile, the only supported operations WILL be the ECF 4.0 ReviewFiling operation and the corresponding synchronous response. It WILL NOT support any of the ECF 4.0 query, asynchronous response, or electronic service operations or the RecordFiling operation.”

Recommendation: Either introduce a new version of the Portable Media specification for ECF v4.1 or prescribe Portable Media SIP compatibility in section 5.3. If the former, adequately address FAMDE-to-FRMDE message exchanges, in addition to Review Filing Request.

10. NotifyDocketingComplete operation optional?

Section 3.1 ‘The Filing-Preparation-to-Docketing Process Model’ specifies that “the operations in bold are required” and then provides a list of all operations. In this list the NotifyDocketingComplete operation is not in bold text.

Figure 4 in the same section shows the NotifyDocketingComplete operation outside of any shaded rectangle labeled as ‘opt’ (understood to mean ‘optional’). As such, the implication provided in Figure 4 is that the NotifyDocketingComplete operation is not optional, hence required.

Recommendation: Correct the specification and remove this ‘optional vs required’ ambiguity. Likely this means to correct Figure 4 so that NotifyDocketingComplete() is included within a shaded ‘opt’ rectangle.

11. Is machine-readable Court Policy mandatory (in an ECF compliant implementation)?

The specification is unclear whether machine-readable Court Policy is required.

a) First off, the term ‘machine-readable court policy’ appears to be defined at the beginning of section 2.4 ‘CourtPolicy’, as:

• **Machine-readable court policy** – an ECF 4.1 message that describes the features of the ECF 4.1 implementation supported by this specification, the court’s code lists and any other information a Filing Assembly MDE would need to know in order to successfully submit an electronic filing into that court.

The ’ECF message’ is presumably CourtPolicyResponseMessage, which would be used for machine-readable Court Policy and never for human-readable Court Policy.

b) Section 3.1 ‘The Filing-Preparation-to-Docketing Process Model’ identifies that the CourtPolicy operation is not required, and the GetPolicy() exchange between FAMDE and FRMDE is optional.

c). Section 3.2.1 ‘GetPolicy’ states “The Filing Assembly MDE MAY obtain a court’s machine-readable court policy at any time by invoking the GetPolicy operation on the Filing Review MDE with the identifier for the court“, and “This step may be omitted if the Filing Assembly MDE already has the current court policy.” The key word here is “may” and not “must”.

d) Nevertheless, on line 506 in section 2.4.2 ‘Machine-Readable’ Court Policy’, says:

“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.”

It is not clear what this requirement means.

Taken all together, it appears to:

a) require machine-readable court policy in some form (not necessarily in design, structure or content as seemingly envisioned by the CourtPolicyQueryMessage, provided that it meets the purposes in section 2.4.2).

b) require that the FAMDE utilize this Court Policy (but not necessarily obtain this court policy via an ECF GetPolicy operation or by employing CourtPolicyResponseMessage). It seems that FAMDE is allowed to either already possess current court policy or must obtain it by invoking the ECF v4.1 GetPolicy operation.

If the above is a correct interpretation, then what is to be made of the words from line 506 “the machine readable court policy”?

Has the hyphen been deliberately left out of “machine readable” so that it is not the same as “machine-readable”?

If the hyphen were to be included, then does line 506 refer to the definition provided on line 453 (including the words “an ECF 4.1 message”)?

If the absence of the hyphen indicates a different definition than that provided on line 453 (such as court policy information, not provided in the form of an ECF 4.1 message) then is the statement on line 506 simply trying to say that Filing Assembly MDEs must have access to some form of computer prescribed policy data?

Recommendation:

Language consistency throughout the specification is not only beneficial, but imperative. Clarify the requirement on line 506, making it clear whether “machine readable” (no hyphen) is the same as “machine-readable” (with hyphen), or by adding the hyphen.

If “machine readable” is intended to be understood as different than “machine-readable” then this should be clearly spelled out.

The phrase “…by some other means” in line 506 implies that Court Policy information could be conveyed via machine-readable word processing documents, text files, spreadsheets, data bases and other means implemented within FAMDEs and thereby not requiring any FRMDE provided machine-readable Court Policy communication (e.g., employing an ECF 4.1 message).

12. Asynchronous response optionality

In the circumstance in which a court does not implement CourtPolicy (e.g., the GetPolicy operation is optionally supported by an FRMDE, nevertheless see #11 above) and as such there will not be any GetPolicyResponse, then how is the requirement for, or optionality of, an asynchronous response determined (e.g., how does an operation determine if and when to provide an asynchronous response)?

First off, since <ecf:SendingLocationID> and <ecf:SendingMDEProfileCode> elements are now optional in all messages where these elements are allowed (e.g., available in all except PaymentMessage and PaymentReceiptMessage), then when these elements are not provided, the operation cannot provide any asynchronous response.

So, in the absence of court policy, when these elements are provided, then is the receiving operation expected to provide an asynchronous response?

Since the ECF v4.1 specification is silent on this, it appears that it is left up to the operation. Is this what the TC intended?

If so, then when machine-readable court policy has not been implemented and a sending MDE provides <ecf:SendingLocationID> and <ecf:SendingMDEProfileCode> then the receiving operation may or may not provide an asynchronous response.

There are only two asynchronous responses, i.e., RecordDocketingCallbackMessage and ReviewFilingCallbackMessage.

Recommendation:

a. Make clear how MDEs are expected or required to act when ECF Court Policy is not implemented, and ReviewFiling requests and RecordFiling requests have been provided that include <ecf:SendingLocationID> and <ecf:SendingMDEProfileCode>. Are the receiving MDEs required to provide the asynchronous response or are they free to not provide the asynchronous response.

b. When ECF Court Policy is not used, then require the Human-Readable Court Policy specify whether asynchronous response messages must or will be provided and any circumstances regarding this expectation.

13. ECF 4.1 Compliant Required Messages and Operations

Section 2.2 Major Design Elements states (line 355): “In order to be compliant with ECF 4.1, an MDE must support all messages required 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.”

The key here appears to be the word “required” as in “required for that MDE”.

Perhaps it is noteworthy that the ECF v4.1 specification never states which messages are required for any MDE (it does identify required operations in section 3.1). Appendix C identifies ECF messages which are request and response parameters for operation calls. However, Appendix C is not normative. Perhaps instead of saying “an MDE must support all messages required for that MDE” it should say “an MDE must support all required operations for that MDE”.

Required ‘messages’ may be able to be deduced from required operations (note that currently there is ambiguity in the specification regarding RecordDocketingCallbackMessage; see #10 above).

By deduction, there are only 2 (or perhaps 3) required message exchanges in ECF 4.1, i.e., ReviewFilingRequest (CoreFilingMessage/PaymentMessage), RecordFilingRequest (RecordDocketingMessage/CoreFilingMessage), and, depending on the resolution of #10 above, NotifyDocketingComplete (RecordDocketingCallbackMessage).

Note the prior issue #10 – NotifyDocketingComplete is likely not a required operation.

Recommendation:

a. Identify required (normative) messages within the specification (possibly in Appendix C). It appears that this is only:

* CoreFilingMessage/PaymentMessage in a ReviewFilingRequest and
* RecordDocketingMessage/CoreFilingMessage in RecordFilingRequest.

b. In the circumstance where an implementation is requiring asynchronous callback messages, then also:

* NotifyDocketingComplete and
* ReviewFilingCallbackMessage (in this circumstance is PaymentReceiptMessage also required? If so, does that make PaymentReceiptMessage a callback message?)

c. and/or, revise the statement “an MDE must support all messages required for that MDE” to instead say “an MDE must support all required operations for that MDE”.

Furthermore, also in section 2.2, it states (line 347): “a complete ECF 4.1 system MUST include at least one each of the Filing Assembly, Filing Review and Court Record MDEs.”

Since the Filing Assembly MDE no longer has any required operations (e.g., NotifyFilingReviewComplete no longer need be provided), then should this provision be revised?

Or is the intent that an MDE (e.g., FAMDE) must not only provide required operations, but also must be able to invoke required operations on other interfacing MDEs (e.g., as illustrated in Figure 4)? If stated this way, the FAMDE could still be a required operation since ReviewFiling is a required operation. Presumably, any software/system that can invoke ReviewFiling is therefore an FAMDE, even if this system is not capable of enabling “a filer to create a filing message for submission to a court”.

Or is the intent that any operation that can be required but may not be required, such as NotifyDocketingComplete and NotifyFilingReviewComplete, must be provided by the appropriate MDE, even when not required by or not employed in an implementation?

Further Recommendation:

d. Consider whether MDEs must provide operations that may be optionally required in an implementation, such as NotifyDocketingComplete and NotifyFilingReviewComplete.

Presumably, operations that are not employed in an implementation also will not need to be supported by relevant MDE. If this is the case, then when NotifyFilingReviewComplete is not implemented, can a ‘complete ECF 4.1 system’ only provide FRMDE and CRMDE? Or, if a ‘complete ECF 4.1 system’ must provide FAMDE (even when not using NotifyFilingReviewComplete), is it a requirement that FAMDE must be able to invoke FilingReview on the FRMDE?

e. Modify the first sentence in Appendix C (line 1337), revising “operations that are provided” to “operations that may be provided”.

14. Additionally, consider the statement in section 3.1 (line 587), i.e., “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.” Recall that as of ECF 4.1 there are now only two required operations, i.e., ReviewFiling and RecordFiling.

It is not clear what is meant by the provision “as long as sending and receiving MDEs are implemented in separate systems”.

I seem to recall that this provision was added as a way to address systems in which the Clerk Review functions of the FRMDE were provided within the same software system (e.g., Case Management System) as the RecordFiling functions. Thereby only making RecordFiling a required operation when the FRMDE and the CRMDE are in separate software systems.

If this is the intent, then take note of the compliance statement in section 2.2 ‘Major Design Elements’, line 347, “a complete ECF 4.1 system MUST include at least one each of the Filing Assembly, Filing Review and Court Record MDEs.” This precludes any system in which the Clerk Review functions are provided within the same software system as docketing/RecordFiling functions, wherein the RecordFiling operation need not be named “RecordFiling” and need not be invoked using ECF 4.1 RecordFilng request XML, from being considered as a “complete system”.

Note that it’s not clear what the distinction is between a “complete system” and a “compliant implementation”. Section 2.2 uses the phrases “compliant implementation” and “complete ECF 4.1 system”, suggesting that ‘complete’ may not be ‘compliant’ and perhaps that ‘compliance’ need not be ‘complete’.

a. Consider the following restatements:

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.

This has the effect of making the RecordFiling operation a “conditionally” required operation.

So, when the RecordFiling function is not explicitly provided in an ECF v4.1 compliant manner (e.g., is provided within the same software system as the Clerk Review functions), and when <RequireAsynchronousResponsesIndicator> in Court Policy is true, then is the callback message (e.g., ReviewFilingCallbackMessage) required?

I would think so, even if it is a misnomer to consider it a callback message.

b. Finally, in Section 2.2, line 355 provides “In order to be compliant with ECF 4.1, an MDE must support all messages required for that MDE.” Should the word “must” be “MUST” instead?

Recommendations:

a. Revise Section 3.1, replacing “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” with “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.”

b. In Section 2.2, revised the word “must” to be all caps, i.e., “MUST”..

15. NotifyDocketingComplete

The presumption made in this feedback item is that RecordDocketingCallbackMessage is optional and therefore the NotifyDocketingComplete operation is also optional (see prior issue #10 above).

In Section 3.2 Business Rules, subsection 3.2.7 NotifyDocketingComplete, the following paragraph is provided:

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 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. 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. The Filing Review MDE responds synchronously with an acknowledgement of the callback message.

Every time I read this, I feel something has been left out or left unsaid. Is this an accidental omission or intentional? Why?

To remove this uncertainty, I propose, at a minimum, the following revision:

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 to indicate whether the filing was accepted or rejected by the court record system, otherwise the asynchronous callback message is optional. If the Court Record MDE rejected the filing, an explanation MUST be provided. 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. The Filing Review MDE responds synchronously with an acknowledgement of the callback message.

Unpacking this section into multiple paragraphs (e.g., one for each significant idea presented) may facilitate readability and understanding. Consider:

RecordDocketingCallbackMessage MAY be provided as a callback message by the Record Filing MDE to the Filing Review MDE to indicate whether the filing was accepted or rejected by the court system. The Filing Review MDE responds synchronously with an acknowledgement of any callback message received.

When the <RequireAsynchronousResponsesIndicator> in the court policy is “true”, the Court Record MDE MUST invoke the NotifyDocketingComplete operation on the Filing Review MDE, otherwise the callback message is optional.

If the Court Record MDE rejected the filing, an explanation MUST be provided in the callback message when provided to Filing Review MDE. 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 when a callback message is tendered.

Note: other additional possible edits are identified in other sections of this document (e.g., #19 c.1, etc.).

Recommendation: Revise section 3.2.7 for clarity and accuracy.

16. NotifyFilingReviewComplete

Section 3.2.8 ‘NotifyFilingReviewComplete’ states:

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 as a callback message to the ReviewFiling operation to indicate whether the filing was accepted and docketed by the clerk and court record system. The operation MAY return the filed documents or links to the documents, but MUST include the [FIPS 180-4] SHA 256 document hash, a condensed representation of a document intended to protect document integrity.

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. The Filing Assembly MDE responds synchronously with an acknowledgement of the callback message.

The first sentence, in the first paragraph is not clear. It appears to prescribe two conditions upon which the callback message is required to be provided to the FAMDE:

1) when the clerk rejects the filings

or

2) when the FRMDE (which provides Court Policy) knows that <RequireAsynchronousResponsesIndicator> in Court Policy is ‘true’ and upon receipt of a RecordDocketingCalbackMessage (misstated as ‘NotifuyDocketingComplete message’).

If I place parenthesis around the conditions as might be done in many programming languages, this sentence looks like:

If (the clerk rejects the filings) or ( (the Filing Review MDE receives the NotifyDocketingComplete message) and (the <RequireAsynchronousResponsesIndicator> in the court policy is “true”) ), then the Filing Review MDE MUST invoke the NotifyFilingReviewComplete operation on the Filing Assembly MDE as a callback message to the ReviewFiling operation to indicate whether the filing was accepted and docketed by the clerk and court record system.

Alternative parentheticals are:

If ( (the clerk rejects the filings) or (the Filing Review MDE receives the NotifyDocketingComplete message) ) and (the <RequireAsynchronousResponsesIndicator> in the court policy is “true”), then the Filing Review MDE MUST invoke the NotifyFilingReviewComplete operation on the Filing Assembly MDE as a callback message to the ReviewFiling operation to indicate whether the filing was accepted and docketed by the clerk and court record system.

Which interpretation is correct is not clear.

There does not appear to be any other statements in the specification, or included technical artifacts, that could be additionally applied to the first sentence in 3.2.8 to resolve this ambiguity.

Additional points:

* There is no such thing as the ‘NotifyDocketingComplete message’ (NotifyDocketingComplete is an operation not a message). However, there is a RecordDocketingCallbackMessage which is provided on the NotifyDocketingComplete operation invocation call.
* Secondly, the messages provided on the invocation call to the NotifyFilingReviewComplete operation are called ReviewFilingCallbackMessage and PaymentReceiptMessage (see C.1 ‘Filing Assembly MDE’, C.1.1 ‘Provided Operations’).
* Since RecordDocketingCallbackMessage is now optional, the results of docketing may not be provided to the Filing Review MDE. As such, it may not be possible for the CRMDE to inform the FRMDE whether the filing(s) were docketed. As such, the words “and docketed’ may need to be removed from the statement “to indicate whether the filing was accepted and docketed by the clerk and court record system”.
* Similarly, since the FRMDE may not know docketing/CRMDE results, the words “and court record system” may need to be removed from the first sentence in the second paragraph.

Recommendation.

Both clarify the section making it clear under which circumstances the invocation of the FAMDE NotifyFilingReviewComplete operation is required, and also correct other misstatements.

For example (e.g., assuming the second parenthetical example is the true intended interpretation):

ReviewFilingCallbackMessage and PaymentReceiptMessage MAY be provided as callback messages by the Review Filing MDE to the Filing Assembly MDE to indicate whether the filings were accepted by the clerk. The Filing Assembly MDE responds synchronously with an acknowledgement of any callback message received.

When the <RequireAsynchronousResponsesIndicator> in the court policy is “true”, the Filing Review MDE MUST invoke the NotifyFilingReviewComplete operation on the Filing Assembly MDE upon receipt of a RecordDocketngCallbackMessage from the Court Record MDE, otherwise the callback message is optional.

The operation MAY return the filed documents or links to the documents but MUST include the [FIPS 180-4] SHA 256 document hash, a condensed representation of a document intended to protect document integrity.

If the filing included a payment, and the filing was accepted by the clerk, a receipt for the payment MUST be included in the operation.

Note: Depending upon the resolution of #17 below, the word “filings” above may instead be “filing”.

17. Inconsistent use of the word ‘filing’.

The word ‘filing’, sometimes is used in the singular form ‘filing’ and at other times is used in the plural form ‘filings’.

Sometimes the word ‘filing’ appears to be used as a synonym for ‘submission’ (e.g., as in an RvFR) and sometimes it is used as a synonym for ‘document’ or ‘document rendition’, or as a “combination of data and documents”.

Examples:

1. In 3.2.8 ‘NotifyFilingReviewComplete’ is says “if the clerk rejects the filings” (plural). It is not clear if “filings” is intended to refer to the submission or the documents provided in a submission. Submission makes more sense in which case it should be singular.

This understanding (as ‘submission’ rather than ‘document’) seems consistent with the use of the word “filing” in the section immediately prior (i.e., 3.2.7 NotifyDocketingComplete). In this section, filing is used in “to indicate whether the filing was accepted or rejected” and “if the Court Record MDE rejected the filing”. Here, it appears ‘filing’ (singular) is intended to be understood as ‘submission’ and not as ‘document’.

At a minimum, the word “filing” should be used consistently in both 3.2.7 and 3.2.8. e.g., in its singular form or plural form. It appears that the singular form is correct. (Also see 3.2.3 ‘GetFeesCalculation’, 3.2.4 ‘ReviewFiling’ 3.2.5 ‘ServeFiling’, 3.2.6 ‘RecordFiling’, 3.2.9 ‘GetFilingList’, 3.2.10 ‘GetFilingStatus’. In each of these sections, the word “filing’ refers to ‘submission’ and not to ‘documents’ provided in a submission.)

2. In the Glossary in section 1.4 ‘Terms and Definitions’, ‘**Filing**’ is defined as: “An electronic document (with any associated data, attachments and the like) that has been assembled for the purpose of being filed into a specified court case.” Whereas this definition may be suitable for a common language or court/legal community usage of the term, it does not appear to be the definition of “filing” as used in Section 3.1 ‘Business Rules’.

Recommendations:

Recognizing that the word “filing”, or “filings” has multiple usages, such as common language usages, legal community business and court usages, as well as ECF technical language usages, then use these terms consistently.

a. Revise 3.2.8 to use the singular “filing” and not “filings”. (These terms appear to have a consistent technical usage within section 3.2 ‘Business Rules’.)

b. Develop visual cues, such a selective capitalization, italics, special fonts, etc. which are consistently applied to different usages of these terms.

c. Review and revise if appropriate, the definitions provided in the Glossary.

18. Element description for CoreFilingMessage

The xsd:documentation element for CoreFilingMessage in ECF-4.1-CoreFilingMessage.xsd reads:

The structure of a Filing including any Payment Information will be documented in this section. This describes the filing transaction between the Filing Assembly MDE and the Filing Review MDE. This information will become part of the Record Docketing between the Filing Review MDE and the Court Record MDE but does not necessarily describe the information that is actually stored in the Court Record.

Since Payment information is not provided in CoreFilingMessage and is provided in the separate PaymentMessage, the description (above) should be corrected.

Furthermore, CoreFilingMessage (CFM) not only appears in the RvFR (ReviewFilingRequest) and RFR (RecordFilingRequest), it also appears in FeeesCalculationQueryMessage. As such, the words “this describes the filing transaction between the Filing Assembly MDE and the Filing Review MDE” are not accurate for all usages for CoreFilingMessage, as it appears to exclude FeesCalculationQueryMessage. In the FeesCalculationQueryMessage, CFM is, at best, only a prospective (e.g., potential) filing.

Recommendation:

Both issues may be corrected by the following rewording:

The structure of a Filing will be documented in this section. This describes the proposed or actual filing transaction between the Filing Assembly MDE and the Filing Review MDE. This information may become part of the Record Docketing between the Filing Review MDE and the Court Record MDE but does not necessarily describe the information that may be actually stored in the Court Record.

19. Typo – line 7 in section 1.1 ‘Scope’

A space character appears to be missing between the words “system” and “and” in “This specification describes the technical architecture and the functional features needed to accomplish a successful electronic court filing system,and defines both the normative (required) and non-normative (optional) business processes it supports.”

Recommendation: Add a space between “…court filing system,” and “add defined both…”, as in “This specification describes the technical architecture and the functional features needed to accomplish a successful electronic court filing system, and defines both the normative (required) and non-normative (optional) business processes it supports.”

20. Cardinality for RecordDocketingCallbackMessage within NotifyDocketingComplete

Even though wrappers.xsd permits unlimited RecordDocketingMessages within a RecordFilingRequest., one RecordDocketingMessage (with a single ReviewedLeadDocument) for each FilingLeadDocument within CoreFilingMessage, only one RecordDocketingCallbackMessage is allowed within a single NotifyDocketingComplete.

A single RecordDocketingCallbackMessage can report on docketing results for a single ReviewedLeadDocument only.

There is no place within the prose specification where the number of RecordDocketingCallbackMessages permitted within a call to the NotifyDocketingComplete operation is ever stated. In section C.2.1 ‘Provided Operation’ for Filing Review MDE within Appendix C. ‘(Informative) MDE Operations’, operation invocation parameters are provided. Note however that this is not normative. To wit, see ECF TC email #436363, Jan. 14, 2014, (<https://www.oasis-open.org/apps/org/workgroup/legalxml-courtfiling/email/archives/201401/msg00005.html>) which raises this issue during the “IJIS Springboard Initiative”.

For NotifyDocketingComplete, the invocation parameter is RecordDocketingCallbackMessage. Whether this parameter is required (or is optional), and how many instances of the specific parameter are allowed is never expressed. As such, for the information presented in the prose specification, it seems, these choices are implementation dependent.

Whenever multiple ReviewedLeadDocuments are present within a single RecordFilingRequest, then either (1) multiple invocations of NotifyDocketingComplete are required (one for each ReviewedLeadDocument) or (2) a single NotifyDocketingComplete request must be allowed to provide multiple RecordDocketingCallbackMessages.

With ECF v4.1, when using wrappers.xsd, only the first option presented above is available, i.e., a CRMDE must make multiple calls to NotifyDocketingComplete, passing a single RecordDocketingCallbackMessage with each invocation, when there are multiple ReviewedLeadDocuments in a submission

In ECF v4.01, and when employing one of ECF 4.01’s various Web Services SIPs, it may not have been apparent to implementers that the WS-SIP intended that only a single RecordDocketingCallbackMessage should be provided within a NotifyDocketingCompleteMessage specified in the wsdl. Web Services implementations based on the ECF-4.0-WebServicesProfile-Definitions.wsdl may not have enforced this restriction, or implementers may have modified the wsdl.

Furthermore, if implementers had chosen to implement using the IBM MQ SIP, then this SIP’s exchange schema (i.e., ECF-4.0-IBMMQProfile-Definitions.xsd) permits multiple RecordDocketingCallbackMessages in a single NotifyDocketingCompleteRequest.

Since the ECF 4.0 (and ECF 4.1 to a certain extent) prose specification does not specify invocation parameters, whether they are required or optional and any parameter cardinality, then both the Web Services SIP and the IBM MQ SIP are equally conformant.

However, ECF v4.1 introduces wrappers.xsd. This schema does specify that a single NotifyDocketingComplete element must contain one and only one NotifyDocketingCompleteRequest element, which in turn, must contain one and only RecordDocketingCallbackMessage.

Yet, ECF v4.1 never specifies that wrappers.xsd must be used, and it does not appear in the ‘Informative’ Appendix C. It appears that implementers are free to use or not use wrappers.xsd as desired. It does appear in the D.1 ‘Operation Invocation’ example in Appendix D ‘(Informative) Example Transmissions’ of the [ECF-WS-SIP-v4.1] specification, if somehow the reader understands that the ‘wrappers’ namespace prefix refers to ‘urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1’.

Recommendations:

1. Determine whether wrappers.xsd truly belongs in the main ECF v4.1 specification, or if it more appropriately belongs in the ECF 4.1 Web Services SIP specification (see #7 above).

2. Presuming the determination is made to continue wrappers.xsd within the main ECF v4.1 specification, then:

a. Define the role of the ‘wrappers’ schema within the specification, not leaving this up to individual implementations. Determine whether the use of wrappers.xsd is required or optional, or SIP dependent. If required, then consider all Service Interaction Profiles.

b. Consider revising Appendix C making it normative and not just informative. Include parameter optionality and cardinality.

c. Allow multiple RecordDocketingCallbackMessage elements within a NotifyDocketingComplete operation invocation request.

c.1 If NotifyDocketingComplete is not revised to permit multiple RecordDocketingCallbackMessages, then revise section 3.2.7 ‘NotifyDocketingComplete’ to require a callback message for each FilingLeadDocument that has been provided on the RecordDocketing request. These multiple NotifyDocketingComplete requests, responding to a single RecordDocketing request, need not all be provided in quick succession (e.g., time gaps, short or long, between callback messages are permitted).

21.Cardinality for ReviewFilingCallbackMessage in NotifyFilingReviewCompleteRequest

Wrappers.xsd only permits a single instance of ReviewFilingCallbackMessage within NotifyFilingReviewCompleteRequest.

A single ReviewFilingCallbackMessage can only provide information for a single ReviewLeadDocument.

A ReviewFilingRequest may provide multiple LeadDocuments.

As such, multiple RevewFilingCalbackMessages may be required for a single submission (e.g., RvFR).

There is no place within the prose specification where the number of ReviewFilingCallbackMessages that are permitted within a call to the NotifyFilingReviewComplete operation is ever stated. In section C.1.1 ‘Provided Operation’ for Filing Assembly MDE within Appendix C. ‘(Informative) MDE Operations’, the operation invocation parameters are provided.

For NotifyFilingReviewComplete, the parameters are ReviewFilingCallbackMessage and PaymentReceiptMessage. Whether an individual parameter is required, and how many instances of a specific parameter are allowed or required, is never expressed. As such, for the information presented in the prose specification, it seems, these choices are implementation dependent.

The exact parameters and parameter cardinality is provided in wrappers.xsd. However, use of wrappers.xsd is not required in the specification.

Recommendations:

1. Determine whether wrappers.xsd truly belongs in the main ECF v4.1 specification, or if it more appropriately belongs in the ECF 4.1 Web Services SIP specification (see #7 above).

2. Presuming the determination is made to continue wrappers.xsd within the main ECF v4.1 specification, then:

a. Define the role of the ‘wrappers’ schema within the specification, not leaving this up to individual implementations. Determine whether the use of wrappers.xsd is required or optional, or SIP dependent. If required, then consider all Service Interaction Profiles.

b. Consider revising Appendix C making it normative and not just informative. Include parameter optionality and cardinality.

c. Allow multiple ReviewFilingCallbackMessage elements within a NotifyFilingReviewCompleteRequest. Note that if NotiyDocketingComplete is revised to allow multiple RecordDocketingCallbackMessage elements (see #20 2.c above) this this revision is essential.

c.1 If NotifyFilingReviewCompleteRequest is not revised to permit multiple ReviewFilingCallbackMessages, then revise section 3.2.8 ‘NotifyFilingReviewComplete’ to require a callback message for each LeadDocument that has been provided on the ReviewFiling request. These multiple NotifyFilingReviewCompleteRequests, responding to a single ReviewFiling request, need not all be provided in quick succession (e.g., time gaps, short or long, between callback messages are permitted).

22. Incorrect version reference.

Appendix A ‘(Informative) Release Notes’, subsection A.2 ‘Package Structure’ line (1210) provides: “The files in these subdirectories, linked to the specification document, contain the various normative and informational pieces of the 1.0 release”.

The reference to ‘the 1.0 release’ appears to be incorrect if the word ‘release’ is to be generally understood as ‘version’.

Recommendation: Correct the statement in Appendix A, subsection A.2 to provide the correct version number.

23. Root folder name

In Appendix A, subsection A.2 ‘Package Structure’ is says: ‘Unzipping this archive creates a directory named ecf-4.1/ containing this specification document and a number of subdirectories.’

In the current Public Review package, the root folder created upon unzipping the ecf-webservices-v4.1-csd01.zip file is ecf-v4.1-csd01.

Perhaps the file and folder names will be as currently provided in section A.2 in the approved Committee Draft version, but it seems likely that the csd01 designation will remain.

As currently provided, this inaccuracy may be disadvantageous to Public Reviewers, especially those who are not ECF TC members.

Recommendation:

a. Consider how these references should be provided through various iterations in the specification drafting, working, review, public review and approval cycle. The TC should consider writing and maintaining a document describing the process for updating specification documents and producing distribution kits (e.g., zip files).

b. Update Appendix A for the appropriate csd01 designators.

24. Depth of Recursive Structures

Recommendation: The advisory that implementations may limit the depth of recursive structures in section A.3 include a recommendation or requirement that any such limit or restriction be included in Court Policy?

25. Subdirectories

Appendix A, subsection A.2 ‘Package Structure’ states (line 1210): “A description of each subdirectory is given below.” Only 3 subdirectories are listed and described. Not included are: ‘ecf-v4.1-csd01\_files’ and ‘model’.

Recommendation: Include the missing subdirectories in the listing in A.2.

26. Record on Appeal

No schema is recommended for or provided for Record on Appeal transactions (presumably this may be ReviewFilingRequest, but it is not stated). No illustrative example XML instances are provided. ECF 5 provides a series of 8 Record on Appeal examples. Perhaps these can be ‘back-revised’ to ECF 4.1.

Recommendation: Identify ECF messages and schema to be used for Record on Appeal transactions. Provide one or more illustrative XML instance examples.

27. Conformance Clause

Section 2.2 ‘Work Product Components’ within the OASIS ‘Technical Committee (TC) Process’ provides:

2.2.6 Conformance Clauses. A Standards Track Work Product that is approved by the TC at the Committee Specification Public Review Draft, Committee Specification or OASIS Standard level must include a separate section, listing a set of numbered conformance clauses, to which any implementation of the specification must adhere in order to claim conformance to the specification (or any optional portion thereof).

A definition for Conformance Clause is provided in ‘Definitions of Key Terms’:

“Conformance clause” is a statement in a Standards Track Work Product that references one or more Normative Statements in order to describe a condition that an implementation must satisfy in order for it to conform to that part of the Work Product.

Presumably, Section 7 ‘Conformance’ has been provided in the ECF v4.1 specification to address this requirement. Section 7 provides:

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.

Other OASIS specifications provide enumerated compliance Rules within the specification’s Conformance section.

For example, from the Code List Representation (Genericode) specification, the following shows the first few conformance rules:

4 Conformance

4.1 Auxiliary Rules

An XML instance conforms to the OASIS Code List Representation genericode document model if it does not violate any constraints expressed in the genericode.xsd schema associated with this version of the specification, including auxiliary rules marked as “document” rules.

An application conforms to the OASIS Code List Representation genericode processing rules if, in addition, it does not violate any of auxiliary rules marked as “application” rules.

4.2 Category: document

Some of these document-related auxiliary rules can be programmatically tested using ISO/IEC 19757-3 Schematron as described in Appendix D, Testing select document conformance rules (Non-Normative).

Rule 1 [document:: complexType CodeListDocument]

A code list must have at least one key, unless it is a metadata-only definition without a 'SimpleCodeList' element.

Rule 4 [document:: element CanonicalUri in complexType CodeListRef]

Must be an absolute URI, must not be relative.

Rule 6 [document:: element CanonicalVersionUri in complexType CodeListRef]

Must be an absolute URI, must not be relative.

Rule 19 [document:: attribute Type in complexType Data]

The datatype ID must not include a namespace prefix.

For the W3C XML Schema datatypes, possible datatype IDs are 'string', 'token', 'boolean', 'decimal', etc.

The Conformance Rules illustrated above for ‘Genericode’ appear to clearly meet the OASIS requirement to provide a set of numbered conformance clauses.

Many other OASIS specifications appear more in line with the approach provided in ECF v4.1. It is not known whether OASIS Admin will ever enforce the requirement to provide a set of numbered conformance clauses such as provided by Genericode, but such an enumeration may be useful in ECF, for ECF implementers and for interoperability.

It is probably not best to hold up approval of the ECF v4.1 specification to develop an enumerated conformance requirements list. Perhaps this is something that can be added in later document revision.

However, I suspect that any effort to provide such a clear, unambiguous conformance enumeration will highlight issues within the specification that may require additional numbered versions to address.

A prior attempt at a requirements enumeration can be accessed from TC email #412822 (12-Feb-2013) using the following link: <https://www.oasis-open.org/apps/org/workgroup/legalxml-courtfiling/email/archives/201302/msg00002.html>

This matrix is for ECF 4.01 (or 4.0.1) and follows a different approach than used by Genericode.

The Springboard project is an additional attempt at identifying and testing for compliance. ‘Springboard’ included Web Services SIP compliance along with main specification compliance.

The following is a listing of the MUST requirement from the ECF v4.1 specification provided for this Public Review:

|  |  |  |
| --- | --- | --- |
| **No.** | **Section/Line #** | **Requirement** |
| 1 | 2.1/322 | 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 | 2.2/346 | 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 | 2.2/349 | 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 ? | 2.2/355 | In order to be compliant with ECF 4.1, an MDE must support all messages required for that MDE. |
| 5 & 6 | 2.4/457 | 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 | 2.4/460 | The court’s human-readable and machine-readable court policies MUST each have a version numbering method associated with it. |
| 8 | 2.4.1/476 | To be compliant with the ECF 4.1 specification, each court MUST publish a human-readable court policy |
| 9 | 2.4.1/477 | human-readable court policy MUST include each of the following:  1. The unique court identifier  2. The location of the machine-readable court policy  3. A definition of what constitutes a “lead document” in the court  4. A description of how filer identifiers are to be maintained during electronic communications regarding the case  5. A description of how the court processes (dockets) filings  6. A description of any instances in which the court will mandate an element that the ECF 4.1 schema makes optional  7. 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 | 2.4.2/506 | 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 | 2.4.5/577 | If court-specific constraint schemas are used, instance documents MUST validate against both the ECF schemas and the court constraint schemas. |
| 12 | 3.1/586 | 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 | 3.2/616 | Successful queries MUST return an <ecf:ErrorCode> of “0”. |
| 14 | 3.2/616 | Failed queries MUST NOT return an <ecf:ErrorCode> of “0” |
| 15 | 3.2.2/630 | the Court Record MDE MUST have access to the court’s registry with all updated information about case participants. |
| 16 | 3.2.2631 | There MUST be only one such registry per court, |
| 17 | 3.2.2/635 | 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. | 3.2.4/650 | The Filing Assembly MDE MUST submit the filing to the court by invoking the ReviewFiling operation on the Filing Review MDE. |
| 19. | 3.2.5/657 | 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. | 3.2.5/662 | 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. | 3.2.5/666 | 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. | 3.2.6/669 | If the clerk reviews and accepts the filing, the Filing Review MDE MUST invoke the RecordFiling operation on the Court Record MDE. |
| 23. | 3.2.7/674 | 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. | 3.2.7/677 | If the Court Record MDE rejected the filing, an explanation MUST be provided. |
| 25. | 3.2.7/677 | 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. | 3.2.8/683 | 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. | 3.2.8/687 | The operation MAY return the filed documents or links to the documents, but MUST include the [FIPS 180-4] SHA 256 document hash, |
| 28. | 3.2.8/690 | 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. | 3.3.1.1/723 | Attachment identifiers MUST be unique within a message transmission. |
| 30. | 3.3.1.2/727 | Case identifiers (case numbers) are assigned by the court record system and MUST be unique within a court. |
| 31. | 3.3.1.3/730 | 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. | 3.3.1.3/732 | Court identifiers MUST conform to following convention:  <Internet domain of the court administrator>:<unique identifier within the court system>. |
| 33. | 3.3.1.4/741 | Document identifiers are assigned by the court record system and MUST be unique within a court. |
| 34. | 3.3.1.5/743 | Filing identifiers MUST be unique within a court and will be generated by the court in response to a ReviewFiling operation. |
| 35. | 3.3.1.6/746 | The address of an MDE MUST be unique within a given communications infrastructure. |
| 36. | 3.3.1.7/750 | 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. | 3.3.1.8/755 | 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. | 3.3.3.1/830 | 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. | 3.3.3.1/835 | If a CoreFilingMessage includes documents, the message MUST include only one level of connected and supporting documents. |
| 40. | 3.4/854 | 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. | 3.4/862 | 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. | 3.4/871 | 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. | 3.4/875 | 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. | 3.4/879 | 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. | 3.4/883 | 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. | 3.4/886 | 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. | 3.4/890 | 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. | 3.4/899 | Individual documents within the ROA transaction MUST not be individually accepted or rejected. |
| 50. | 3.4/900 | All documents within the ROA transaction MUST have the same acceptance or rejection disposition. |
|  |  |  |

Recommendation:

1. Provide clear, unambiguous conformance enumeration that highlight issues within the specification that may require additional numbered versions to address.
2. Do not hold up approval of the ECF v4.1 specification to develop an enumerated conformance requirements list.
3. Add an enumerated conformance requirements list in later document revisions.