**Core vs SIP Regarding core specification cardinality for NotifyDocketingComplete and NotifyFilingReviewComplete (Log #26 & 27):**

At the April 3, 2023, ECF TC meeting, when reviewing Issues #26 and #27 from within the ECF 4.1 Public Feedback log, it was agreed to table these two items to allow time for further consideration. These feedback items addressed cardinality of ECF messages in NotifyDocketingComplete and NotifyFilingReviewComplete.

The key point requiring additional consideration is whether MDE operation invocation signatures should be normatively stated in the core specification or in SIP specifications.

J. Cabral believes that it is both okay and prudent to have the cardinalities regarding the ECF messages provided in an operation invocation to be specified by the SIP and not in the core spec. As such, the cardinality may be different in one SIP than in another. He also does not want to make Appendix C in the core spec. normative.

Presently, in ECF 4.1 (and in prior OASIS ECF versions, e.g., ECF 3) the parameters provided for an operation invocation (e.g., operation signatures/contracts) are not normatively defined in the core ECF specification. The core specification does provide an informative (i.e., non-normative) Appendix C. Appendix C lists operations, both required and optional, provided by the four MDE. Each operation identifies ECF message invocation parameters and ECF message return type. For invocation parameters, the optionality and cardinality for individual parameters are not provided.

It is noteworthy that in the core ECF specification, within section 5 ‘Service Interaction Profiles’, the following statement is provided: “A service interaction profile does not govern the content of messages – message content is described in Sections 2 and 3 of this specification“.

Understanding this statement depends on just exactly what is meant by “message”, as well as “content of message”. If “message” is used in the sense of a SOAP message then it appears to be inconsistent with the position that operations signatures are specified in SIP. If, however, “message” is used in a narrower sense, meaning one of the ECF defined messages, then there may not be any inconsistency. The use of the word message in this context should be clarified, such as replacing “message” with “ECF message”.

In the sentence immediately prior to the one provided above, i.e., “An ECF 4.1 service interaction profile defines a transmission system that supports the functional requirements of electronic filing, along with the MDE operations and message structures, and implements certain non-functional requirements” the words “message structures” is included. In this wording, does the word “message” have the same meaning as in the very next sentence?

I must presume that the word “message” has the same meaning in both usages within the same paragraph.

So, if “message” means one of the ECF messages, then “message structures” may mean structures that assemble or aggregate ECF messages (such as request structures and response structures, e.g., an exchange structure). Note that wrappers.xsd performs this same ‘message structure’ function for ECF 4.1, yet it is in the core specification (allowed but not required) and is not provided by any SIP.

Section 5 ‘Service Interaction Profiles’ of the core specification, lays out the purpose and requirements for Service Interaction Profiles. This section does say that a SIP defines “message structures”. Once again, it is not clear what is meant by “message structures”.

If defining “message structures” is a key facet of a SIP, then why is there not an enumerated requirement for the SIP to define “message structures” listed within section 5.1 ‘Service Interaction Profile Requirements’?

The use of the word “message” gets more convoluted following the two usages in the first paragraph provided above. In section 5.1 ‘Service Interaction Profile Requirements’, the word “message” appears to be more synonymous with a SOAP message or perhaps a “message structure’.

Adopting the viewpoint that the structure of exchanges (e.g., operation invocation parameters/signatures) are defined by the SIP, is a departure.

Regardless of various possible interpretations of specifications and words such as “message”, “message structure” or “message content”, etc., this viewpoint appears to be new. Up until now, all variations in SIPs have, as their fundamental basis, differing exchange transmission technologies, and not differing “exchange structures” or both.

Proceeding in this direction, then at some point, there may be multiple Web Services SIPs for a given core ECF specification (e.g., v4.1). Such as, one for “regular” efiling and another for “bulk” efiling (see ‘Batch/Bulk Filing Myth’ in the ‘Batch/Bulk Filing Sidebar’ below).

To a certain degree, having multiple TC approved SIPs for a given ECF version is not new. For example, there are multiple ECF TC recommended Web Services SIPs for ECF 4.0 and 4.01, such as the 3 listed as replaced or superseded in the [ECF-WS-SIP-v4.1] Committee Specification Draft 01 document, i.e., Web Services Messaging Profile 1.0 Specification, Web Services Service Interaction Profile 1.1 Specification and Web Services Service Interaction Profile 1.1 Specification.

However, these older WS SIPs differ in technological choices, and not in exchange structure or operation signature choices. For example, WS SIP v2.0 (Committee Draft, Oct. 2008) and WS SIP v2.01 (Committee Draft, Oct. 1, 2011), differ chiefly in that the older version requires WS-ReliableMessaging Version 1.0, whereas the more recent WS SIP requires WS-MessageReliability Version 1.1. This is a communications technology difference and not a difference in exchange message structures or operation signatures.

To consider a departure as significant as suggested will require careful review and consideration. Multiple specifications and parts of specifications may require revision and enhancement.

Interoperability must be considered.

Let’s say there are two Web Services SIPs that are both TC recommendations for ECF 4.1. In one (e.g., WS-SIP-A) the NotifyDocketingComplete operation invocation only allows a single RecordDocketingCallbackMessage within a single NotifyDocketingComplete (as specified in wrappers.xsd). In a second TC recommended WS SIP (e.g., WS-SIP-B) multiple RecordDocketingCallbackMessages are allowed within a single NotifyDocketingComplete, otherwise these two WS SIPs are the same.

Now imagine an FRMDE that provides the NotifyDocketingComplete operation. To be ECF 4.1 interoperable, it may need to support both WS-SIP-A and WS-SIP-B. For the single, simple difference imagined between these two example SIPs. Although supporting both may be plausible, it would be additional effort and expense. Hopefully, after implementing the FRMDE that supports both WS-SIP-A and WS-SIP-B, the ECF TC does not then approve WS-SIP-C that specifies a third, different operation signature for NDC.

Consider however, that if WS-SIP-A supported ReviewFiling as specified by wrappers.xsd, and another WS-SIP-B allowed bulk/batch efiling, permitting unlimited combinations of CoreFilingMessages and PaymentMessages, then interoperability will be strained.

By allowing the message exchange structures to be defined within the SIP then this has the effect of allowing the operation invocation signature to be defined outside of the core ECF specification. As such, and whereas the operations themselves and the expected operation behavior may still be established within the core specification, the operation calling signature would be outside of the core specification.

If the above is the intent, then what if a SIP wanted to define the RecordDocketing call signature as including both the currently prescribed RecordDocketingMessage and CoreFilingMessage as well as the existing PaymentMessage and not just the RecordDocketingMessage + CoreFilingMessage as informatively (but not normatively) described in the core specification, section C.3.1 (shown below):

|  |  |  |  |
| --- | --- | --- | --- |
| **Operation** | **Called By** | **Output** | **Parameters** |
| RecordFiling | Filing Review MDE | xsd/message/ECF-4.1-MessageReceiptMessage.xsd : MessageReceiptMessage | xsd/message/ECF-4.1-RecordDocketingMessage.xsd : RecordDocketingMessage |
| xsd/message/ECF-4.14.0-CoreFilingMessage.xsd : CoreFilingMessage |

Example 1

Would this be a new SIP, or an extension to an existing SIP or an extension to the core specification?

Since, as it appears, it could be any of these, it seems to throw interoperability out of the window.

But also consider that in the core specification, Section 2.2 ‘Major Design Elements’ it claims “The MDEs defined in the ECF 4.1 specifications are meant only to define the “interface” to each operation; the specification is not intended to define how operations must be implemented“. This statement appears to be in contrast to the point of view put forward, if “interface” is understood to be the operation invocation signature. This statement appears to state that the operation signature is the provenance of the core specification.

I do understand that a proposal for some type of radically different or challenging SIP (perhaps due to unusual or unanticipated exchange message structures) would need to be submitted to, reviewed by, then approved by the ECF TC, thereby some control may be exercised.

But consider that ECF does not truly have conformance criteria, e.g., there is no real, practical means to determine or establish that an implementation is conformant. Sure, technically, Section 7 in the core specification may satisfy the OASIS requirement to provide Conformance Requirements, but simply referring to sections 1 -6 is not complete, clear or truly measurable. Within the referenced section (i.e., 2.2), oddly placed beneath Legal Service MDE, additional compliance criteria is expressed, i.e., “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.“ Additionally, this section also requires that “In order to be compliant with ECF 4.1, an MDE must support all messages required for that MDE” (note this will be revised to say “all required operations for that MDE” (see Feedback Change Log #24)).

An unscrupulous implementer could provide an ECF compliant FRMDE that supports “all required operations for that MDE”, including the ReviewFiling operation, but still not support CoreFilingMessage in the invocation of the ReviewFiling operation and still be compliant.

Of course, it’s hard to understand how such an implementation of ReviewFiling would ever be useful.

I suppose it bears repeating that I do understand that a proposal for some type of radically different or challenging SIP (perhaps due to unusual or unanticipated exchange message structures) would need to be submitted to, reviewed by, then approved by the ECF TC, thereby some control may be exercised.

Note however, that in the core ECF specification there does not appear to be any requirement that any SIP (including ECF TC recommended SIPs) need ever be employed in an ECF compliant implementation (e.g., a SIP-less ECF implementation can be a compliant implementation).

Maybe this is intended. And perhaps this could be advantageous when an implementation wants to define and use its own locally defined SIP (or proprietary API) and never seek an ECF TC recommendation.

Returning to Section 2.2 ‘Major Design Elements’ in the core specification wherein it claims “The MDEs defined in the ECF 4.1 specifications are meant only to define the “interface” to each operation; the specification is not intended to define how operations must be implemented“. With regard to the agreed revision (Log #24) to say “an MDE must support all required operations for that MDE”, it still seems inadequate to not itemize MDE required ECF messages in normative fashion, somewhere within the core ECF specification.

In not doing so, this would appear to leave this as the provenance of the SIP.

If it is correct that operation signatures are defined in SIP and not in the core specification, then there should be tables in the SIP specifications that are similar to those provided in the core specification in Appendix C. These SIP provided tables should also specify the order in which parameters must be provided (if important), the parameter optionality, and parameter cardinality.

For Example:

The Filing Assembly MDE provides the following operations to other MDEs:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Operation** | **Called By** | **Output** | **Parameters** | **Order** | **Optional or Required** | **Max Allowed** |
| NotifyFilingReviewComplete | Filing Review MDE | xsd/message/ECF-4.1-MessageReceiptMessage.xsd : MessageReceiptMessage | xsd/message/ECF-4.1-ReviewFilingCallbackMessage.xsd : ReviewFilingCallbackMessage | 1 | Required | Unlimited |
| xsd/message/ECF-4.1-PaymentReceiptMessage.xsd : PaymentReceiptMessage | 2 | Optional | 1 |

Example 2

Consider that in the above, values provided for the column labeled ‘Optional or Required’ may be complicated by requirements or allowances established in the core specification.

Suppose a SIP did allow unlimited ReviewFilingCallbackMessages within a NotifyFilingReviewComplete invocation, but also wanted to constrain/limit this to circumstances in which there are multiple ReviewedLeadDocuments within a single RecordFilingRequest (e.g., a single ‘filing’ under a single filing identifier).

Where is this additional constraint expressed? Does it belong in the SIP? The answer may be obvious; the core specification is fixed (for given version), so any additional parameter restrictions or constraints must be expressed in the SIP.

If there is an existing SIP that would serve the purpose of the imagined implementation, sans the additional constraint, then to include this additional constraint within the existing SIP would require what? A new SIP? Or a SIP extension? At present, there does not appear to be any ECF endorsed approach to SIP extension.

Consider, for a Web Services SIP, whether the division into WSDLs is a significant factor in the SIP. WSSIP for ECF v4.1 is now split into 4 WSDL, one for each MDE. This is a departure from prior WSSIPs that provided a single WSDL for all 4 MDEs.

Is this significant?

For starters, note that the prose WS SIP specification document lists these four WSDL under ‘Additional Artifacts’.

Presumably, if there were a Web Services SIP, that provided separate WSDL for each operation, but was otherwise identical to [ECF WS-SIP-v4.1], e.g., same technologies, etc., then this would be a different SIP.

The following table appears to describe the separate purpose and responsibilities between the core specification and the SIP from the perspective that operation signatures are the responsibility of the SIP (the table is not intended to be a complete enumeration, but is limited to those items relevant to this discussion, controversial items in red text):

Demarcation: purpose and responsibility.

|  |  |
| --- | --- |
| **Core** | **SIP** |
| * Defines ECF messages and ECF message content. * Provides suggested but optional message/exchange structures (wrappers.xsd). * Defines operations (name and purpose as well as ECF requirements and restrictions in function). * Defines groupings of operations into MDEs. | * Defines exchange structures composed of ECF messages. * Defines operation signatures. * Can define additional operations requirements and restrictions. * For Web Services SIP, specifies the quantity and scope of WSDL. |

Bullet 2 above is in blue text, intended to highlight that, even though wrappers.xsd provides structures for operation signatures, these are not normative in the core specification. If the TC should decide that operation signature definitions belong in the core specification and not in SIP, then the ‘optional’ use of wrappers.xsd may be revised.

So, if the core specification prescribed operation signature parameters (e.g., using wrappers.xsd), then how might the table entry in Appendix C of the core specification appear?

Would it appear as:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operation** | **Called By** | **Output** | **Parameters** |
| RecordFiling | Filing Review MDE | xsd/message/ECF-4.1-MessageReceiptMessage.xsd : MessageReceiptMessage | xsd/wrappers.xsd : RecordFilingRequest |

Example 3

Or as:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operation** | **Called By** | **Output** | **Parameters** |
| RecordFiling | Filing Review MDE | xsd/message/ECF-4.1-MessageReceiptMessage.xsd : MessageReceiptMessage | xsd/message/ECF-4.1-RecordDocketingMessage.xsd : RecordDocketingMessage |
| xsd/message/ECF-4.14.0-CoreFilingMessage.xsd : CoreFilingMessage |

Example 4

Or perhaps as:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operation** | **Called By** | **Output** | **Parameters** | |
| RecordFiling | Filing Review MDE | xsd/message/ECF-4.1-MessageReceiptMessage.xsd : MessageReceiptMessage | xsd/wrappers.xsd : RecordFilingRequest | xsd/message/ECF-4.1-RecordDocketingMessage.xsd : RecordDocketingMessage |
|  | xsd/message/ECF-4.14.0-CoreFilingMessage.xsd : CoreFilingMessage |

Example 5

Recommendations:

1. TC must determine whether MDE operation invocation signatures are normatively defined in the core specification or are to be normatively established within the SIP.

In making this decision, impacts on interoperability, the quantity and scope of SIPS, and other impacts should be considered.

My personal view is that operation signatures should be normatively defined within the core specification as this better promotes interoperability.

2. Whatever decision is made, then it should be made clear in both the core specification and in each SIP specification. A table demarcating the purpose and responsibility division may be appropriate in each specification. At a minimum, this information should be provided in Section 5, ‘Service Interaction Profiles’ in the core specification.

2.a. If it has been decided that SIPs are charged with defining operation invocation signatures:

2.a.1 then add this decision into Section 5.1, ‘Service Interaction Profile Requirements’, clarifying that a key distinguishing facet of a Service Interaction Profile is that it defines “message structures” (“although the term “message structures” may need revision or clarification).

2.a.2 In each SIP specification document, add a normative section that defines the SIP required operation invocation signature (see Example 2 provided). The operation signature must specify each ECF message parameter, the order in parameters must be provided, whether the parameter is optional or required, and the cardinality for each parameter.

2.a.3 Review the core specification for use of the words ‘request’ and ‘response’. Since, with this decision, the exact technical specification of request structures and response structures are defined in SIP, then the core specification cannot assign a precise definition to these terms. For example, the term review filing request cannot be presumed to consist of one CoreFilingMessage and (optionally) one PaymentMessage.

2.a.4 Define and document an ECF endorsed approach or methods for SIP extension.

2.a.5 For Web Services SIPs, determine whether a new SIP is required to further subdivide SIP provided WSDL (e.g., one for each operation) into any other division (e.g., one for each operation).

2.b. If instead it is decided that the core specification is charged with defining normative operation signatures, then

2.b.1 add this decision into the core specification (appropriate section and wording to be determined).

2.b.2 Make Appendix C normative. Add additional operation signature information into Appendix C that establish and clarify parameter sequence, optionality and cardinality.

3. Clarify the various usages of the words/terms “message”, “message content, “message structure, etc. Glossary updates should be anticipated.

4. Regarding Batch or Bulk filing:

4.a Revise the core specification to remove the misleading mention of ‘multiple (batch) filings’ in Section 2.4.2 ‘Machine-Readable Court Policy’.

4.b. In the Web Services SIP specification for v4.1, remove the erroneous statement in the Introduction regarding the added support for bulk filings.

Or, in the alternative

4.c Properly add batch/bulk filing into the specifications, defining exchange structures for batch/bulk filing, rules, conditions, restrictions, etc.

**Batch/Bulk Filing Sidebar** – the Batch/Bulk Filing Myth

In section 2.4.2 ‘Machine-Readable Court Policy’ it mentions ‘batch filings’. This is the only time ‘batch filings’ is mentioned in the core specification (it may be mentioned in the Web Services SIP 4.1 specification if “bulk filing” is the same as “batch filing”).

There are no schemas or WSDL that specify the construction of a batch filing (or bulk filing). For an ReviewFilingRequest, one might envision a single large transaction containing multiple CoreFilingMessages and associated PaymentMessage(s) (e.g., multiple ReviewFilingRequests).

One might even image that in a ‘batch filing’, a single submitter may provide multiple CoreFilingMessages for the very same court and case, but only provide a single payment for all filing fees and other fees (e.g., a single PaymentMessage for multiple CoreFilingMessages).

Furthermore, the Web Services SIP v4.1 says (Introduction) “This version adds support for bulk filings”. At present, the ECF TC has no such SIP. Batch and/or bulk filings appears to be a myth.

If there was a SIP that supported ‘batch filings’ then should Human-Readable Policy also state which SIP to use for ‘batch filings’ as well as any other filing?

Given that there was a ‘batch filing’ SIP that permitted multiple CoreFilingMessages and multiple PaymentMessages, and allowed a single PaymentMessage for multiple CoreFilingMessages, then how should the table in C.2.1 in the core specification be interpreted?

C.2.1 **Provided Operations**

The Filing Review MDE provides the following operations to other MDEs:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operation** | **Called By** | **Output** | **Parameters** |
| ReviewFiling | Filing Assembly MDE | xsd/message/ECF-4.1-MessageReceiptMessage.xsd : MessageReceiptMessage | xsd/message/ECF-4.1-CoreFilingMessage.xsd : CoreFilingMessage |
| xsd/message/ECF-4.1-PaymentMessage.xsd : PaymentMessage |

Should this table be understood as specifying that the ReviewFiling operation takes exactly two parameters (e.g., one and only one CoreFilingMessage and one and only one PaymentMessage) or is it suggesting that when invoking ReviewFiling, at least one CoreFilingMessage may be optionally provided and at least one PaymentMessage may be optionally provided, and both multiple CoreFilingMessages and multiple paymentsMessages may be provided as well (e.g., as specified by SIP)?

Recall that feedback from the ICJIS Springboard project addressed this very section/tables, in feedback item I-8 shown below:

I-8. **Incorrect Parameter Reference**

Issue Details:

• The provided operations tables in sections C.1.1, C.2.1 and C.3.1 each contain one instance of an operation that is defined with two “Parameters”. The table entries, however, do not clearly identify whether the parameters are both optional, both required or conditional.

• After reviewing the web service definition, we discovered that the actual base parameters are defined in the WSDL and extend the “ElectronicFilingMessageType” type. As such, the parameters column, depicted below, does not seem to properly represent the actual web service definition

Springboard provided the following recommendations regarding I-8:

Suggestion:

• Update the tables in sections C.1.1, C.2.1 and C.3.1 to properly reflect the base types.

• Additionally, consider moving the full type declaration from the WSDL to a message schema.

Note that second suggestion to consider moving full type declaration (e.g., message structure) from WSDL to message schema appears to be what has been done by adding wrappers.xsd.