**ECF 4.01 Web Services SIP 2.01 Review**

From Appendix B. (Informative) Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Editor** | **Changes Made** |
| Wd01 | 2022-06-18 | James Cabral | Changes to ECF 4.01 Web Services SIP 2.01:  Split the previous WSDL into separate files for each MDE; changed the WSDLs to use document literals, (aligning operation names and root elements) and include a SOAP action in the binding; fixed reference to MTOM specification. |

“Split previous WSDL into separate files for each MDE”

* The previous wsdl (v2.01) was ECF-4.0-WebServicesProfile-Definition.wsdl

The new wsdl’s are:

* CourtRecordMDE.wsdl
* FilingAssembly.wsdl
* FilingReviewMDE.wsdl
* ServiceMDE.wsdl

Feedback:

Let’s start off easy:

1. There appears to be a typo in Appendix C: (Informative) Example Implementation. The following is included in the Example WSDL:

xmlns:wsmp="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecorMDE-4.1"

It looks like the letter “d” is missing in CourtRecordMDE-4.1, as shown below:

targetNamespace="urn:oasis:names:tc:legalxml-courtfiling:schema:wsdl:CourtRecordMDE-4.1"

1. The web link below is reported as ‘Not Found’:

[WS-I BP1.1]

K. Ballinger, D. Ehnebuske, C. Ferris, M. Gudgin, M. Nottingham, C. K. Liu, P. Yendluri, *Basic Profile Version 1.1*, <http://www.ws-i.org/profiles/basicprofile-1.1-2004-08-24.html>, WS-I Organization, August 2004.

1. Messages

In the ECF 4.1 specification, the word “message’ has a very specific and limited meaning as defined in section 2.3.1, which states:

A message is an XML document that is a well-formed XML data structure with a single root element that is transmitted between MDEs and is valid as defined by one of the defined message structure schemas in the ECF 4.0 specification.

The term “message” in the ECF specification may not be identical to the use of this term in other contexts, such as SOAP, however, it should be the same as used in ECF WS SIP (see section 1.3 Terms and Definitions). Observe that an operation input ‘message’ defined in WSDL, such as tns:ReviewFilingRequest, may not be an ECF message. A ReviewFilingRequest contains two ECF messages, CoreFilingMessage and PaymentMessage.

The words “defined by one of the message structure schemas in the ECF 4.0 specification” is understood to refer to section 4.4 ECF 4.0 Message Schemas. This section list 24 specific schema. None of these is ReviewFilingRequestMessage schema. The schema in this section does include CoreFilingMessage schema and PaymentMessage schema.

Additionally, Appendix D (Informative) Example Instances, lists 25 example messages, including CoreFilingMessage and PaymentMessage, but not ReviewFilingRequestMessage.

Furthermore, section C.2.1 Provided Operations, lists two Parameters for the ReviewFiling operation, specifically, CoreFilingMessage and PaymentMessage.

XMLSpy has a feature to create a SOAP Request (SOAP > Create SOAP Request …). If you use this feature with the Example implementation wsdl files (e.g., CourtRecordMDE-ImplementationExample.wsdl), then . . .

The result for the GetCase operation is:

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

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:m0="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CaseQueryMessage-4.0" xmlns:m1="http://niem.gov/niem/structures/2.0" xmlns:m2="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CommonTypes-4.0" xmlns:m3="http://niem.gov/niem/niem-core/2.0" xmlns:m4="http://niem.gov/niem/domains/jxdm/4.0" xmlns:m5="http://niem.gov/niem/ansi-nist/2.0">

<SOAP-ENV:Body>

<m:GetCase xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m0:CaseQueryMessage m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

<m2:SendingMDELocationID m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

<m3:IdentificationID m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1"/>

Observe that the above SOAP does conform to 2.4 Operation Addressing, i.e.,

Each message transmission MUST either identify the operation or operations being invoked or be a synchronous response to a previous request. Each operation MUST be either a REQUIRED operation as defined in the ECF 4.0 specification or an OPTIONAL operation identified as supported by the court through the current machine-readable court policy. The response to a request for an operation not supported by the court MUST be reported using the ECF 4.0 <ErrorCode> element in the core message and MAY also include a SOAPFault in the SOAP envelope.

For ReviewFiling, XMLSpy generates:

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

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:m0="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CoreFilingMessage-4.0" xmlns:m1="http://niem.gov/niem/structures/2.0" xmlns:m2="http://niem.gov/niem/niem-core/2.0" xmlns:m3="http://niem.gov/niem/domains/jxdm/4.0" xmlns:m4="http://niem.gov/niem/ansi-nist/2.0" xmlns:m5="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CommonTypes-4.0" xmlns:m6="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:PaymentMessage-4.0" xmlns:m7="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2" xmlns:m8="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2">

<SOAP-ENV:Body>

<m:ReviewFiling xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m:ReviewFilingRequestMessage>

<m0:CoreFilingMessage m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

<m2:DocumentApplicationName m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1" applicationVersionText="String"/>

Two things can be observed:

1. The second element beneath the SOAP Body element is misnamed as m:ReviewFilingRequestMessage (it should just be m:ReviewFilingRequest) and,
2. The element levels between the SOAP Body element and the ‘true’ message element (e.g., CaseQueryMessage) are inconsistent.

The GetCase has two levels (i.e., GetCase & CaseQueryMessage):

<SOAP-ENV:Body>

<m:GetCase xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m0:CaseQueryMessage m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

Whereas ReviewFiling has three levels (ReviewFiling, ReviewFilingRequestMessage, & CoreFilingMessage/PaymentMessage):

<SOAP-ENV:Body>

<m:ReviewFiling xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m:ReviewFilingRequestMessage>

<m0:CoreFilingMessage m1:id="ID\_1" m1:metadata="ID\_1"

The RPC style is defined in the style attribute of the soap:binding element:

<binding name='EndpointInterfaceBinding' type='tns:EndpointInterface'>

<soap:binding style='rpc' transport='http://schemas.xmlsoap.org/soap/http'/>

<operation name='echo'>

<soap:operation soapAction=''/>

<input>

<soap:body namespace='http://org.jboss.ws/samples/jsr181pojo' use='literal'/>

</input>

<output>

<soap:body namespace='http://org.jboss.ws/samples/jsr181pojo' use='literal'/>

</output>

</operation>

</binding>

The new WS SIP 4.1 defines the style as “document” (from FilingReviewMDE.wsdl:

<!-- 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-4.1\GetFeesCalculation"/>

<input>

<soap:body use="literal"/>

</input>

<output>

<soap:body use="literal"/>

</output>

</operation>:

See: [WSDL binding styles - IBM Documentation](https://www.ibm.com/docs/en/baw/19.x?topic=files-wsdl-binding-styles)

[Which style of WSDL should I use? - IBM Developer](https://developer.ibm.com/articles/ws-whichwsdl/)

There are 5 WSDL styles/use models /patterns

1. RPC/encoded
2. RPC/literal
3. Document/encoded
4. Document/literal
5. Document/literal wrapped

From prior ECF TC discussions, it was agreed to follow the Document/literal wrapped pattern. However, I cannot find any notation or statement in the WS SIP specification that states this. This seems important to include.

Some sources claim that certain styles (i.e., RPC/encoded and RPC/literal) are not WS-I compliant (see [Which style of WSDL should I use? - IBM Developer](https://developer.ibm.com/articles/ws-whichwsdl/)). The ECF WS SIP specification ‘implies’ WS-I compliance by listing [WS-I BP1.1] in section 1.5 Normative References and by the inclusion of section 1.2.6 WS-I Basic Profile 1.1 (i.e., “Compliance with the requirements of the **[WS-I BP 1.1]**, with the exceptions noted in Section 1.2.7, is REQUIRED for compliance with this service interaction profile”).

WSDL experts, such as Russel Butek ([Which style of WSDL should I use? - IBM Developer](https://developer.ibm.com/articles/ws-whichwsdl/)) may well understand that the ECF WS SIP specification precludes RPC/encoded, RPC/literal, and Document/encoded styles, however, in my view it would be useful to the average specification reader if these conclusions were clearly laid out.

Furthermore, presuming that both ‘Document/literal’ options are WS-I compliant (note: Russel Butek says that “Document/literal is WS-I compliant, but with restrictions”), unless we intend for implementers to choose any of the remaining available choices, the preference, or perhaps the requirement, for ‘Document/literal wrapped’ pattern should be explicitly expressed.

other points to consider:

1. The WS SIP specification requires, in section 2.5 Request and Invocation, that “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.”
2. According to Russel Butek, only with the RPC/encoded style that “the operation name appears in the message, so the receiver has an easy time dispatching this message to the implementation of the operation.” Ex:

<soap:envelope>

<soap:body>

<myMethod>

<x xsi:type="xsd:int">5</x>

<y xsi:type="xsd:float">5.0</y>

</myMethod>

</soap:body>

</soap:envelope>

From (highlight added):

<portType name="PT">

<operation name="myMethod">

<input message="myMethodRequest"/>

<output message="empty"/>

</operation>

</portType>

1. For the Document/literal wrapped pattern (shown immediately below),

<soap:envelope>

<soap:body>

<myMethod>

<x>5</x>

<y>5.0</y>

</myMethod>

</soap:body>

</soap:envelope>

Russel Butek provides (highlight added):

“Notice that this SOAP message looks remarkably like the RPC/literal SOAP message in [Listing 5](https://developer.ibm.com/articles/ws-whichwsdl/#listing-5-rpc-literal-soap-message-for-mymethod). You might say it looks exactly like the RPC/literal SOAP message, but there's a subtle difference. In the RPC/literal SOAP message, the <myMethod> child of <soap:body> was the name of the operation. In the document/literal wrapped SOAP message, the <myMethod> clause is the name of the wrapper element which the single input message's part refers to. It just so happens that one of the characteristics of the wrapped pattern is that the name of the input element is the same as the name of the operation. This pattern is a sly way of putting the operation name back into the SOAP message.”

1. Presumably, the Document/literal wrapped pattern complies with the requirement of WS SIP section 2.5 Request and Invocation. To reach this understanding, a lot depends upon the interpretation of the words “(qualified) operation name” and “qualified name of the first child element of the SOAP body element” as used in the referenced section. How is it determined what is to be “qualified”?

The SOAP message for the FilingReviewRequest generated by XMLSpy for the WS SIP v4.1 WD01 is:

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

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:m0="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CoreFilingMessage-4.0" xmlns:m1="http://niem.gov/niem/structures/2.0" xmlns:m2="http://niem.gov/niem/niem-core/2.0" xmlns:m3="http://niem.gov/niem/domains/jxdm/4.0" xmlns:m4="http://niem.gov/niem/ansi-nist/2.0" xmlns:m5="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CommonTypes-4.0" xmlns:m6="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:PaymentMessage-4.0" xmlns:m7="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2" xmlns:m8="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2">

<SOAP-ENV:Body>

<m:ReviewFiling xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m:ReviewFilingRequestMessage>

<m0:CoreFilingMessage m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

Observe that, overlooking namespace, the SOAP body root element is the ‘name’ of the operation (slyly). Also observe that the name of its immediate child element is ‘ReviewFilingRequestMessage’.

Shouldn’t ‘ReviewFilingRequestMessage’ just be ‘ReviewFilingRequest’? (refer to the definition of ‘message’ in the ECF 4.1 specification, section 2.3.1 (provided above).

To achieve the above (e.g., “ReviewFilingRequest” instead of “ReviewFilingRequestMessage”), changes must be made to wrappers.xsd:

This version of wrappers.xsd has been modified from the version delivered in WS-SIP WD01.

The changes are:

1. Changed element ref to "ReviewFilingRequest"

<xsd:complexType name="ReviewFilingRequestType">

<xsd:sequence>

<xsd:element ref="ReviewFilingRequestMessage"/>

</xsd:sequence>

</xsd:complexType>

AND:

2. changed element name to "ReviewFilingRequest"

<xsd:element name="ReviewFilingRequestMessage" type="ReviewFilingRequestMessageType"/>

Results of SOAP message generation:

<SOAP-ENV:Body>

<m:ReviewFiling xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m:ReviewFilingRequest>

<m0:CoreFilingMessage m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

However, after making the changes above, now element “ReviewFilingRequest” should be of type “ReviewFilingRequestType” and not ‘ReviewFilingRequestMessageType”. This type name change cascades culminating in:

<xsd:complexType name="ReviewFilingRequestType"> <!-- name was "ReviewFilingRequestMessageType" -->

<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"/> <!-- name was "ReviewFilingRequestMessage -->

<!-- changed type (above) to "ReviewFilingRequestType" from "ReviewFilingRequestMessageType" -->

<xsd:complexType name="ReviewFilingType"> <!-- name was "ReviewFilingRequestType" -->

<xsd:sequence>

<xsd:element ref="ReviewFilingRequest"/> <!-- was "ReviewFilingRequestMessage" -->

</xsd:sequence>

</xsd:complexType>

<xsd:element name="ReviewFiling" type="ReviewFilingType"/> <!-- changed type from "ReviewFilingRequestType" -->

Undoubtably, similar name pattern changes will also be required for RecordFiling, NotifyDocketingCoimplete, and NotifyFilingReviewComplete.

With the above changes made, XMLSpy generates the ReviewFiling operation SOAP message as:

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

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:m0="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CoreFilingMessage-4.0" xmlns:m1="http://niem.gov/niem/structures/2.0" xmlns:m2="http://niem.gov/niem/niem-core/2.0" xmlns:m3="http://niem.gov/niem/domains/jxdm/4.0" xmlns:m4="http://niem.gov/niem/ansi-nist/2.0" xmlns:m5="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CommonTypes-4.0" xmlns:m6="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:PaymentMessage-4.0" xmlns:m7="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2" xmlns:m8="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2">

<SOAP-ENV:Body>

<m:ReviewFiling xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m:ReviewFilingRequest>

<m0:CoreFilingMessage m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

I believe this is correct (preferred namespace alias mnemonics aside).

All these revisions are within wrappers.xsd which is a part of the main ECF 4.1 specification and not in the WS SIP 4.1 specification. The wrappers.xsd is functioning as NIEM NDR ‘exchange’ schema.

For ReviewFiling and RecordFiling, a three-layered request SOAP message is generated.

For other requests, only a two-layered request is generated:

<SOAP-ENV:Body>

<m:GetFeesCalculation xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m0:FeesCalculationQueryMessage m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

Responses are also two-layered:

<SOAP-ENV:Body>

<m:ReviewFilingResponse xmlns:m="urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:MessageWrappers-4.1">

<m0:MessageReceiptMessage m1:id="ID\_1" m1:metadata="ID\_1" m1:linkMetadata="ID\_1">

e.g., for ReviewFiling, RecordFiling, NotifyDocketingComplete and NotifyFilingReviewComplete:

Operation Name (WSDL) Operation

Request Name (wrapper) Request or Response

Message(s) Name (aka Parameters, message) Message (aka Parameter)

For other requests:

Operation Name (WSDL)

Message Name (aka Parameters, message)

For synchronous responses:

Response Name (wrapper)

Message Name (message)

Do we want ‘balance‘ in layers or is it okay that some exchanges have three layers while others have only two?

4. Are the Supported Service Interaction Profiles, listed in section 5.3 of the ECF v4.1 specification, correct?

For example, it lists:

**Web Services Service Interaction Profile 2.0 Specification**

**Web Services Service Interaction Profile 2.1 Specification**

**Portable Media Service Interaction Profile 1.01 Specification**

It does not include:

**Electronic Court Filing 4.1 Web Services Service Interaction Profile**

It seems that with the changes made in Web Services Interaction Profile 4.1, especially with the addition of and dependency on wrappers.xsd, that WSSIP Profiles 2.0 and 2.1 are no longer compatible.

5. Section B.4 Spreadsheet Models refers to ECF 4.0 (e.g., “ECF 4.0 uses spreadsheet models …”). Should these be revised to just refer to ECF 4 in general? (e.g., “ECF 4 uses spreadsheet models…”).

In the WS SIP 4.1 specification, revisions to ‘Related Work’ and ‘Abstract’ are needed to correct ECF v 4.0 to v 4.1:

Related work:

This specification replaces or supersedes:

* [Web Services Messaging Profile 1.0 Specification](http://docs.oasis-open.org/legalxml-courtfiling/specs/ecf/v3.0/ecf-v3.0-webservices-spec/)
* [Web Services Service Interaction Profile 1.1 Specification](http://www.oasis-open.org/committees/download.php/29417/ecf-v3.1-webservices-spec-cd01.zip)
* [Web Services Service Interaction Profile 2.01 Specification](https://docs.oasis-open.org/legalxml-courtfiling/specs/ecf/v4.0/ecf-v4.0-webservices-spec/v2.01/csprd01/ecf-v4.0-webservices-spec-v2.01-csprd01.zip)

This specification is related to:

* [Electronic Court Filing Version 4.0](http://docs.oasis-open.org/legalxml-courtfiling/specs/ecf/v4.0/)
* WSDL documents: [ECF-4.0-WebServicesProfile-Definitions.wsdl](file:///C:\Users\ggraham\Documents\ECF-4\ECF-4.1\ECF-4.0-WebServicesProfile-Definitions.wsdl),  
  [ECF-4.0-WebServicesProfile-ImplementationExample.wsdl](file:///C:\Users\ggraham\Documents\ECF-4\ECF-4.1\ECF-4.0-WebServicesProfile-ImplementationExample.wsdl)

(Or should the items listed as “This specification is related to:” to move up under “this specification replaces or supersedes”)

And:

Abstract:

This document defines a Service Interaction Profile, as defined in section 5 of the LegalXML Electronic Court Filing 4.0 (ECF 4.0) specification. The Web Services Service Interaction Profile may be used to transmit ECF 4.0 messages between Internet-connected systems.

WS SIP, section 1.1 Relationship to ECF 4.0 Specifications, update to ‘ECF 4.1’, or generalize to ‘ECF 4’.

## Relationship to ECF 4.0 Specifications

The ECF 4.0 specification describes the technical architecture and the functional features of an electronic court filing system, that is, features needed to accomplish electronic filing in a court, pointing out both normative (required) and non-normative (optional) business processes it supports. The non-functional requirements associated with electronic filing transactions, and actions and services needed to accomplish the transactions, such as network structures and security infrastructures, are defined in related specifications, namely:

* Service interaction profile specifications defining communications infrastructures within which electronic filing transactions can take place.
* Document signature profile specifications that define mechanisms for stating or proving that a person signed a particular document.

This specification represents an ECF 4.0 service interaction profile based on web-services. It is intended for implementation in conjunction with the ECF 4.0 specification and at least one ECF 4.0 document signature profile specification. Specifically, in this service interaction profile, the implementation details for each of the Major Design Elements (MDEs), operations, and messages defined in the ECF 4.0 specification, are defined in Web Services Description Language (WSDL).

Section 1.2.3 1.2.3 W3C Simple Object Access Protocol (SOAP) 1.1, make the type of message clear by adding ‘SOAP’, e.g.:

The W3C SOAP 1.1 ([SOAP 1.1]) specification defines SOAP message exchange patterns and SOAP message structures for use with XML. Compliance with the requirements of the SOAP 1.1 specification is REQUIRED for compliance with this service interaction profile.

Even in the referenced SOAP 1.1 specification, great care is taken to distinguish the type of message referred to (e.g., SOAP message or HTTP message):

Following is the response message containing the HTTP message with the SOAP message as the payload:

Note this clarification was provided in WS SIP 2.10 Message Non-repudiation:

The SOAP message MAY include a digital signature applied to the SOA Body and all MIME parts that contain messages or attachments.

6. WS SIP 4.1, 2.1 Service Interaction Profile Identifier, update ECF 4.0 to ECF 4.1:

Each ECF 4.0 service interaction profile MUST be identified with a unique URI which is used in the ECF 4.0 court policy to identify the service interaction profile(s) that a given MDE supports. The ECF 4.1 Web Services Service Interaction Profile will be identified by the following URI:

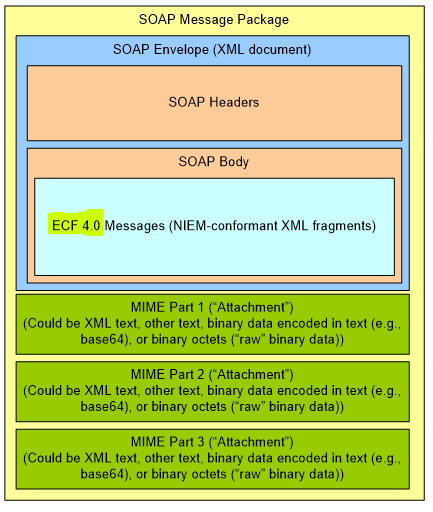
urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:WebServices-4.1

All ECF 4.0 messages sent via this service interaction profile MUST include this URI in the <SendingMDEProfileCode> element. In addition, any court supporting this service interaction profile MUST include this URI in the <SupportedMessageProfile> element in the **CourtFilingResponseMessage**.

2.2 Transport Protocol, update ECF 4.0 to ECF 4.1:

Each ECF 4.0 message transmission sent using this service interaction profile MUST be encapsulated in a SOAP message over the HTTP 1.1 protocol as defined in the **[WSI-I BP 1.1]** and **[SOAP MTOM]** specifications. Figure 1 illustrates the containment of ECF 4.0 messages and attachments within a SOAP Message Package. For compliance with this specification, a SOAP envelope MUST contain one or more messages and MAY contain one or more attachments.

Figure 1. SOAP Envelope with ECF 4.0 Messages and Attachments

. 

WS SIP, 2.7 Asynchronous Mode Response, change ‘MUST’ to ‘MAY’ or ‘MUST’ when <SendingMDELocationID> and <SendingMDEProfileCode> are provided on the request:

The receiving MDE MUST deliver the asynchronous response to a request sent using the web services service interaction profile by sending the asynchronous response to the sending MDE via the web services service interaction profile. The response message transmission MUST conform to the rules for message transmissions established in section 2.5 of this specification above.

2.8 Message/Attachment Delimiters, update ECF 4.0 to ECF 4.1:

The ECF 4.0 messages MUST be encapsulated in the SOAP Body. All other attachments MUST be included in separate MIME parts as shown in Figure 1. The delimiters between the message and the first attachment, and between attachments, MUST comply with the rules for delimiting MIME parts as defined in **[RFC2045].**

7. The messages addressed in 2.9 Message Identifiers, are these RFC 822 messages (as defined in [RFC2045])? In the context of WS SIP 4.21, are these SOAP messages? If so, then it would be clearer if it said so.

8. Are the messages addressed in 2.11 Message Integrity, SOAP messages? If so, then it would be clearer if it said so.

9. Are the messages addressed in 2.11 Message Integrity, ECF messages? If so, then it would be clearer if it said so.

10. Are the messages referred to in 2.14 Message Reliability and 2.15 Message Splitting and Assembly SOAP messages? If so, then it would be clearer if it said so.

11. WS SIP – D.1 Operation Invocation, the example is incorrect. Due to feedback provided in this document, the exact correct form for the example may not yet be determined. But as a minimum, the example is incorrect in that an element with the operation name is not the immediate child element of the SOAP env:body element. Also, in my view, the next element should be wrappers:ReviewwFilingRequest and not wrappers:ReviewFilingRequestMessage (it’s not a message).

12. WS SIP D.2 Synchronous Response example, may need correction pending outcome of wrappers element name discussions.

13. WS SIP D.3 Asynchronous Response example, may need correction pending outcome of wrappers element name discussions.