MQ based Service Interaction Profile (MQSIP)

Abstract

ECF v.4.0 specification has Service Interaction Profiles as one of its three principal elements. The specification offers by default Web Services and Portable profiles. The document discusses yet another - an MQ based Service Profile (MQSIP).

Contents

Problem statement

The state

ECF v.4.0 specification states [lines: 259-271] it contains 3 principal elements:

1. Core specification;
2. **Service Interaction Profiles (SIP);**
3. Document Signature Profiles.

The spec offers two SIPs: Web Services and Portable. It does not prohibit one to create additional profiles.

The document targets the **(2)** in the above list - Service Interaction Profiles part and only, discussing the key decisions behind yet another profile, called here Message Queue (MQ) based SIP -MQSIP.

While the primary ECF TC prescribed task for this doc is to present samples of the MQSIP, we think it would be hard to analyze the samples alone. We've decided to provide some background as well, some explanation to the MQSIP itself - next to the samples requested.

MQSIP profile - top level view

The suggested SIP follows a minimalistic path:

- what is to relay;
- how to relay it.

It takes ECF Core messages for the '*what*' to relay and adds one of the following technologies for the '*how*' to do it:

- Java Messaging Service (JMS), from J2EE;
- Advanced Queue Messaging Protocol (AMQP), from OASIS;
- IBM WebSphere MQ API.

MQSIP targets an enterprise that already has one of the mentioned MQ based technologies in place or would like to introduce one. MQSIP lacks the wealth of many options one can find with WS stack of specifications. MQSIP instead narrows down to the actual basics needed, simplifying the message exchange and implementation(s). Many of the missing from WS stack features can be expected from the MQ provider chosen, e.g.: routing, reliable delivery, security, addressing - as opposed to the corresponding WS-* specs.

Notes

- w3.org Web Services stack of specifications theoretically allows binding to various transports, thus making it possible for example, to use SOAP over TCP/IP with JMS API. Such a solution however would still fall under ECF Web Services SIP category.
  MQSIP on the other hand does not rely on WS specs; it assumes a Service standing behind MDE, same as ECF WS SIP, but not a **Web** Service.
- Intresys has implemented MQSIP for JMS and IBM MQ; these are in production for about 2 years as of time of writing. Intresys has not yet implemented MQSIP with OASIS AMQP.
- MQSIP is not exclusive; can be concurrent with other profiles.

MQSIP - the key design decisions made

Compared to ECF WS SIP in brief:

- remove SOAP and WS stack of specs;
- relax document transfer requirements;
- use the wrapped style to define transport messages;
- create XML schema (*.xsd), no need for WSDL.

Details:

1. Do not use SOAP protocol, envelop.

The primary reasons are:

- MQ based systems have already a number of built-in features, e.g.: routing, reliability/clustering/balancing, security, large binary objects delivery.
  There is no need for a WS stack spec to be implemented to duplicate the features, because those come from MQ provider already.
- SOAP headers lose their usefulness once placed inside an MQ message.
  Once inside a body - SOAP headers are not headers anymore.
  A SOAP envelop itself becomes an extra unused set of bytes in a message.

2. Do not enforce particular way of sending documents (attachments).

- Each MQ provider offers a number of ways to relay large (MBytes) binary objects, e.g.: group of messages, FTP correlated transfer and others , even a streaming (video) objects.
- no Base64 encoding, nor attachments techniques are needed anymore - send document bytes as they are with no overhead.

3. Use the wrapped style when defining message exchange, same as in ECF WS SIP.

- As an illustration.
  A RecordFiling request message in the ECF RecordFiling operation from FR MDE to CR MDE is to become a wrap - RecordFilingRequest (RFR) message - an XML root to contain other ECF messages:

  RecordFilingRequest:
  RecordDocketingMessage,
  CoreFilingMessage,
  other ECF messages.

4. Create XML schema

- The schema is needed to validate sent/received MQ messages, same as in ECF WS SIP.
- No WSDL (Web Service description) file is needed, because we do not employ WS anymore, unlike ECF WS SIP.

Sample flow, schema and messages

Next we'll use few major ECF operations to illustrate what/how needs to be done in order to get to MQSIP.

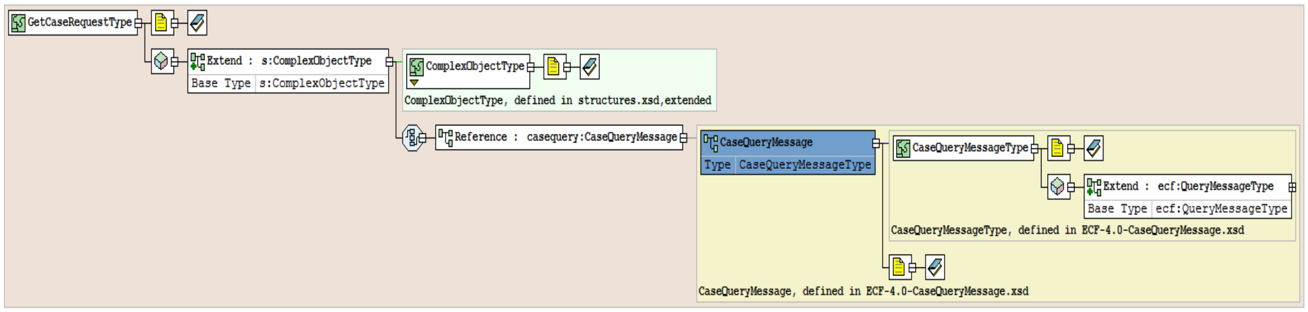
GetCase/GetCaseList/GetDocument operations

Flow:

1. Take a sample GetCase request from ECF v.4.0 distribution (other operations are similar).
2. Wrap it into a GetCaseRequest XML message.
3. Place the result as an MQ text message body and submit using MQ provider.
4. On the way back, receive text GetCaseResponse wrapped message.

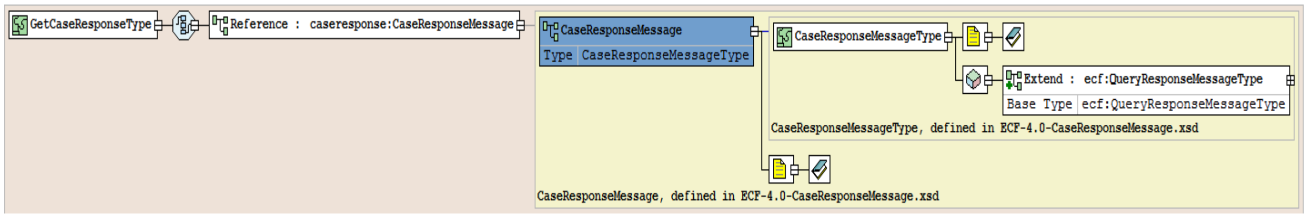
This will hold on to the ECF Core specification v.4.0 using however a different transport - the MQSIP in a nutshell.

Schema illustrations:


GetCaseRequestType
Extend : s:ComplexObjectType
Base Type s:ComplexObjectType
ComplexObjectType
ComplexObjectType, defined in structures.xsd,extended
Reference : casequery:CaseQueryMessage
CaseQueryMessage
Type CaseQueryMessageType
CaseQueryMessageType
Extend : ecf:QueryMessageType
Base Type ecf:QueryMessageType
CaseQueryMessageType, defined in ECF-4.0-CaseQueryMessage.xsd
CaseQueryMessage, defined in ECF-4.0-CaseQueryMessage.xsd

The **GetCaseRequestType** XML schema element defined in MQSIP schema is to provide/validate a **GetCaseRequest** wrapped message. A GetCaseRequest message inside will contain an ECF's **CaseQueryMessage,** as defined by ECF Core v.4.0 specification [lines: 629-632], [lines: 862, 863].

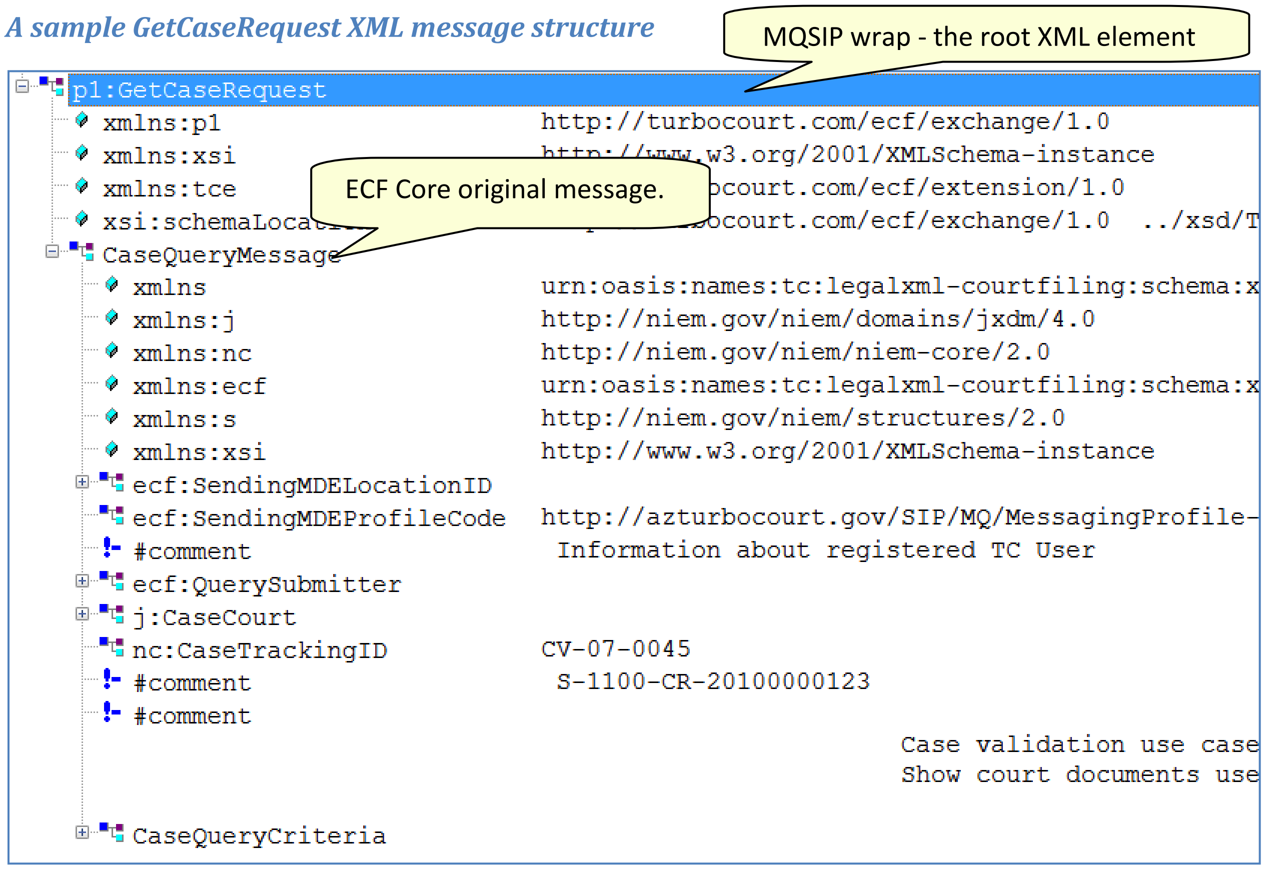
We also need to define a corresponding response message - the **GetCaseResponseType**. This is very similar to the request part of the operation - to contain ECF's **CaseResponseMessage**:


GetCaseResponseType
Reference : caseresponse:CaseResponseMessage
CaseResponseMessage
Type CaseResponseMessageType
CaseResponseMessageType
Extend : ecf:QueryResponseMessageType
Base Type ecf:QueryResponseMessageType
CaseResponseMessageType, defined in ECF-4.0-CaseResponseMessage.xsd
CaseResponseMessage, defined in ECF-4.0-CaseResponseMessage.xsd

That is:

- an MQSIP GetCaseRequest contains ECF's CaseQueryMessage.
- an MQSIP GetCaseResponse contains ECF's CaseResponseMessage.

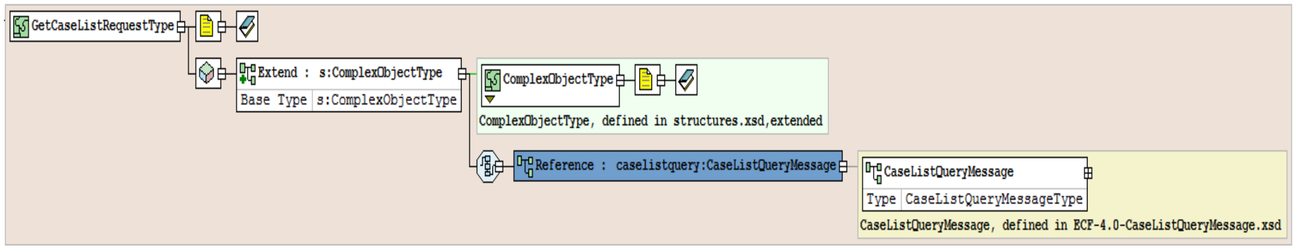
A sample GetCaseRequest XML message structure



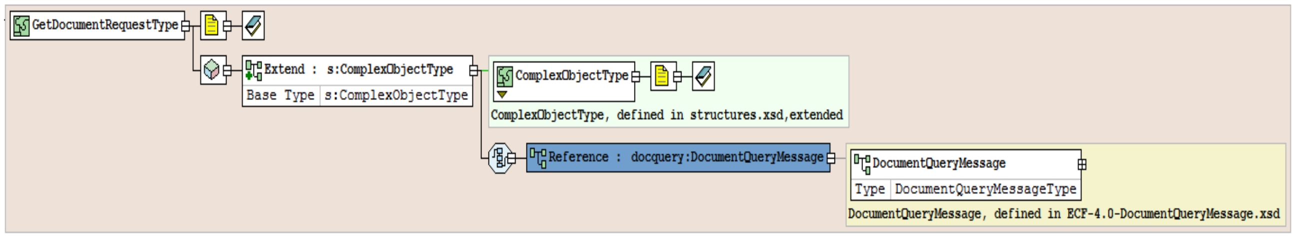
```
p1:GetCaseRequest
    xmlns:p1                    http://turbocourt.com/ecf/exchange/1.0
    xmlns:xsi                   http://www.w3.org/2001/XMLSchema-instance
    xmlns:tce                          ocourt.com/ecf/extension/1.0
    xsi:schemaLoca                     ocourt.com/ecf/exchange/1.0   ../xsd/T
    CaseQueryMessage
        xmlns                   urn:oasis:names:tc:legalxml-courtfiling:schema:x
        xmlns:j                 http://niem.gov/niem/domains/jxdm/4.0
        xmlns:nc                http://niem.gov/niem/niem-core/2.0
        xmlns:ecf               urn:oasis:names:tc:legalxml-courtfiling:schema:x
        xmlns:s                 http://niem.gov/niem/structures/2.0
        xmlns:xsi               http://www.w3.org/2001/XMLSchema-instance
        ecf:SendingMDELocationID
        ecf:SendingMDEProfileCode   http://azturbocourt.gov/SIP/MQ/MessagingProfile-
        #comment                 Information about registered TC User
        ecf:QuerySubmitter
        j:CaseCourt
        nc:CaseTrackingID       CV-07-0045
        #comment                 S-1100-CR-20100000123
        #comment
                                                       Case validation use case
                                                       Show court documents use

        CaseQueryCriteria
```

Other related operations (GetCaseList, GetDocument), as additional illustrations.

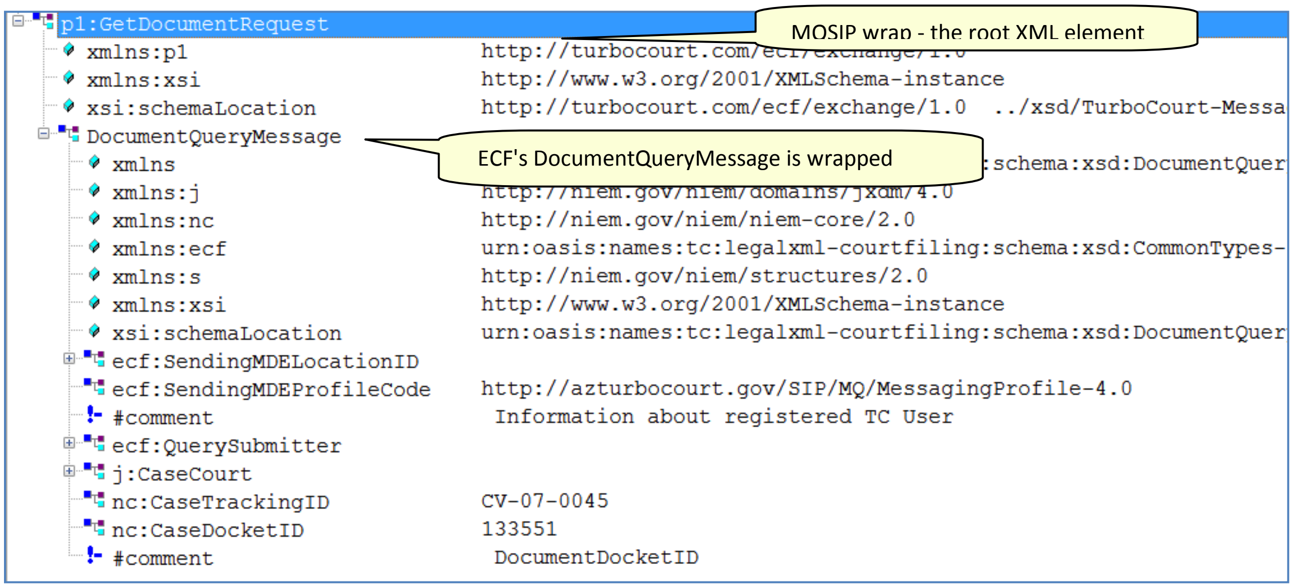

GetCaseListRequestType
Extend : s:ComplexObjectType
Base Type s:ComplexObjectType
ComplexObjectType
ComplexObjectType, defined in structures.xsd,extended
Reference : caselistquery:CaseListQueryMessage
CaseListQueryMessage
Type CaseListQueryMessageType
CaseListQueryMessage, defined in ECF-4.0-CaseListQueryMessage.xsd

As can be seen from the diagram above the CaseListRequestType XML schema element is a wrap around ECF's CaseListQueryMessage. A CaseListRequest message is of CaseListRequestType.


GetDocumentRequestType
Extend : s:ComplexObjectType
Base Type s:ComplexObjectType
ComplexObjectType
ComplexObjectType, defined in structures.xsd,extended
Reference : docquery:DocumentQueryMessage
DocumentQueryMessage
Type DocumentQueryMessageType
DocumentQueryMessage, defined in ECF-4.0-DocumentQueryMessage.xsd

Similar to GetCaseList, the GetDocumentRequestType schema element shown above is a wrap around ECF's DocumentQueryMessage. A GetDocumentRequest message is of GetDocumentRequestType.

A GetDocumentRequest MQSIP message structure looks like follows:


MQSIP wrap - the root XML element
ECF's DocumentQueryMessage is wrapped

```
p1:GetDocumentRequest
  xmlns:p1                     http://turbocourt.com/ecf/exchange/1.0
  xmlns:xsi                    http://www.w3.org/2001/XMLSchema-instance
  xsi:schemaLocation           http://turbocourt.com/ecf/exchange/1.0  ../xsd/TurboCourt-Messa
  DocumentQueryMessage
    xmlns                      :schema:xsd:DocumentQuer
    xmlns:j                    http://niem.gov/niem/domains/jxdm/4.0
    xmlns:nc                   http://niem.gov/niem/niem-core/2.0
    xmlns:ecf                  urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:CommonTypes-
    xmlns:s                    http://niem.gov/niem/structures/2.0
    xmlns:xsi                  http://www.w3.org/2001/XMLSchema-instance
    xsi:schemaLocation         urn:oasis:names:tc:legalxml-courtfiling:schema:xsd:DocumentQuer
    ecf:SendingMDELocationID
    ecf:SendingMDEProfileCode  http://azturbocourt.gov/SIP/MQ/MessagingProfile-4.0
    #comment                    Information about registered TC User
    ecf:QuerySubmitter
    j:CaseCourt
    nc:CaseTrackingID          CV-07-0045
    nc:CaseDocketID            133551
    #comment                    DocumentDocketID
```

Record/Review Filing operations - sending documents

One of the MQSIP's goal is to relax document transfer. It means:

a) start with what MQ providers in use have in common to the participating MDEs;
b) leave the choices the MDEs had agreed to stick to.

For example, practically all JMS providers support group of messages allowing document transfer. Particular MDEs however may negotiate to use correlated FTP transfers.

An MDE can:

- find out the receiving MDE profile to communicate with, which is the same way as in ECF spec - using the MDE profile element.
- detect which is the agreed way to send documents, either from GetPolicy response or from an apriori agreed contract.

Conclusion

The major requested samples have been provided and explained

Compliancy

MQSIP is a compliant ECF Service Interaction Profile:

- it supports all the MDEs
- it uses the corresponding message exchanges from ECF Core specification.

Targets/Efficiency

MQSIP targets enterprise environments with MQ based infrastructure either already in place or those planning to have it. It is believed that MQSIP can be more efficient in MQ based deployments.

Non-exclusiveness

MQSIP does not prevent other ECF service profiles to be present concurrently.
As an example, the ECF RecordFiling operation could be implemented with MQSIP, while an ECF GetCase operation might work fine with ECF's Web Services Profile - all within the same enterprise.

Your comments/critics/suggestions are welcome.
Thank you.

Nov 25, 2012

Serguei Mysko
Solution Architect at Intresys
phone: 805 284 8133
sergueim@intresys.com