IEC TS 62325-504

Edition 1.0 2015-05

TECHNICAL SPECIFICATION

colour inside

Framework for energy market communications – Part 504: Utilization of web services for electronic data interchanges on the European energy market for electricity

IEC TS 62325-504:2015-05(en)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

IEC Catalogue - webstore.iec.ch/catalogue

The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - www.iec.ch/searchpub

The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in 15 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary

More than 60 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

IEC TS 62325-504

Edition 1.0 2015-05

TECHNICAL SPECIFICATION

Framework for energy market communications – Part 504: Utilization of web services for electronic data interchanges on the European energy market for electricity

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 33.200

ISBN 978-2-8322-2694-0

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

FRAMEWORK FOR ENERGY MARKET COMMUNICATIONS –

Part 504: Utilization of web services for electronic data interchanges on the European energy market for electricity

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

The main task of IEC technical committees is to prepare International Standards. In exceptional circumstances, a technical committee may propose the publication of a technical specification when:

- the required support cannot be obtained for the publication of an International Standard, despite repeated efforts, or
- the subject is still under technical development or where, for any other reason, there is the future but no immediate possibility of an agreement on an International Standard.

Technical specifications are subject to review within three years of publication to decide whether they can be transformed into International Standards.

IEC 62325-504, which is a technical specification, has been prepared by IEC technical committee 57: Power systems management and associated information exchange.

The text of this technical specification is based on the following documents:

Enquiry draft	Report on voting
57/1520/DTS	57/1567/RVC

Full information on the voting for the approval of this technical specification can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

A list of all parts in the IEC 62325 series, published under the general title *Framework for energy market communications*, can be found on the IEC website.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC website under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- transformed into an International standard,
- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

FRAMEWORK FOR ENERGY MARKET COMMUNICATIONS –

Part 504: Utilization of web services for electronic data interchanges on the European energy market for electricity

1 Scope

This part of IEC 62325, which is a technical specification, defines the services needed to support the electronic data interchanges between different actors on the European Energy Market for Electricity (EME) in a fast (near-realtime), and secure way. At the same time, this Technical Specification can also be applied to integration problems outside the scope of IEC 62325-451, such as to the integration of gas market systems or general enterprise integration.

Web Services (in WSDL) will be specified for the defined services, applying the Basic Web Service Pattern implementation profile from IEC 61968-100.

The services needed to support the electronic data interchange on the European energy market for electricity are:

- List Messages. This service is used by a client application identified with the credentials of an EME Actor to request a list of messages available on the server for retrieval.
- Get Message. This service is used by a client application identified with the credentials of an EME Actor to request a specific message available on the server.
- Put Message. This service is used by a client application to send a message, usually providing data related to a Market Participant in the energy market for electricity, to the server for processing.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61968-100, *Application integration at electric utilities – System interfaces for distribution management – Part 100: Implementation profiles*

IEC 62325-451-1, *Framework for energy market communications – Part 451-1: Acknowledgement business process and contextual model for CIM European market*

ISO/IEC 40210, *Information technology – W3C SOAP Version 1.2 Part 1: Messaging Framework* (Second Edition)

WSDL, *Web Services Description Language (WSDL) 1.1*

XML Schema 1.0, XML Schema Language Part 1: Structure, W3C Recommendation 28 October 2004; XML Schema Language Part 2: Data Types, W3C Recommendation 28 October 2004

XML Signature Syntax and Processing (Second Edition) http://www.w3.org/TR/xmldsig-core

RFC 6176, *Prohibiting SSL 2.0* http://tools.ietf.org/html/rfc6176

RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* http://tools.ietf.org/rfc/rfc5280

RFC 6818, *Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* http://tools.ietf.org/rfc/rfc6818

RFC 4346, *The Transport Layer Security (TLS) Protocol V1.1* http://www.ietf.org/rfc/rfc4346

3 Terms, definitions and namespaces

For the purposes of this document, the following terms and definitions apply.

3.1 Terms and definitions

3.1.1
message identification
alphanumeric string that represents the name of a message in the system

3.1.2
version
number that represents the message version

Note 1 to entry: The range of values is from 1 to 999.

3.1.3
application time interval
time interval when the message payload applies

3.1.4
server timestamp
date when the message is received by the server (input messages) or when it is made available by the server (output messages).

3.1.5
message type
type of the message payload as defined in IEC 62325-451-n (Schedule_MarketDocument, Acknowledgement_MarketDocument, etc.)

Note 1 to entry: As a general rule the message type is the local name of xml root element.

3.1.6
message code
number identifying a message in the server in a unique way

Note 1 to entry: For a given pair of message codes “x” and “y”, if “y” > “x” then “y” is a newer message. If “y” < “x” then “y” is an older message. Finally if “y” = “x”, then both messages are the same.

3.1.7
data owner
person or entity that is responsible for the information contained in the message (payload)

Note 1 to entry: Usually corresponds with the sender_MarketParticipant.mRID field in the IEC 62325-451-n series.

3.1.8
data provider
person or entity that is responsible for establishing a connection with the server and sending the message (payload)

3.1.9
M/O/C
Mandatory / Optional / Choice (choose one)

Note 1 to entry: Cn indicates “Choice n” and if several optional attributes have the same number “n” in Cn, it means all of them shall be present if this is the selected choice.

3.1.10
status
indication of the acceptance or validity of a message as per 61968-100

Note 1 to entry: The status of messages without Acknowledgement (publication or incoming message still being processed) will be “OK”.

Note 2 to entry: In exchanges where IEC 62325-451-1 applies, the “fully accepted reason code” is associated with a status “OK”. The rest of the reason codes are associated with “FAILED”.

3.2 Namespaces

This Technical Specification uses these prefixes and namespaces (see XML Schema 1.0 Parts 1 and 2):

a) cmsg (urn:iec62325.504:messages:1:0): The target namespace of the messages defined in this Technical Specification.

b) wss (urn:iec62325.504:wss:1:0): The WSDL target namespace for this Technical Specification.

This Technical Specification refers to these other prefixes and namespaces:

a) wsdl (http://schemas.xmlsoap.org/wsdl): This contains the W3C WSDL 1.1 schema.

b) xs (http://www.w3.org/2001/XMLSchema): This contains the W3C XML Schema definition.

c) soap (http://schemas.xmlsoap.org/wsdl/soap): This contains the W3C SOAP bindings for WSDL 1.1.

d) soap12 (http://schemas.xmlsoap.org/wsdl/soap12): This contains the W3C SOAP bindings for WSDL 1.2 (see ISO/IEC 40210).

e) ds (http://www.w3.org/2000/09/xmldsig#): This contains the XML Digital Signature Schema definitions (see XML Signature Syntax and Processing 2nd Edition).

f) msg (http://www.iec.ch/TC57/2008/schema/message): This contains the IEC 61968-100 schema definitions.

4 Conformance

4.1 General

This clause specifies the conformance requirements for a client application and a server to conform to this Technical Specification.

4.2 Client application conformance

In order to conform to this Technical Specification a client application shall:

a) Support the following services as a client:
   - List Messages and all of the mandatory aspects of this service as specified in Clause 5
   - Get Message and all of the mandatory aspects of this service as specified in Clause 5
   - Put Message and all of the mandatory aspects of this service as specified in Clause 5.

b) Send and receive XML Instance documents according to the XML Schema specified in Clause 7 in this Technical Specification for the services listed in a).

c) Use the WSDL definitions, SOAP bindings, and operations specified in Clauses 8 and 9.

d) Be able to access the server via HTTPS, using a client digital certificate recognized by the server for the purposes of establishing the https communication and creating the digital signature as specified in Clause 10.

4.3 Server conformance

In order to conform to this Technical Specification a server shall:

a) Support the following services as a server:
   - List Messages and all of the mandatory aspects of this service as specified in Clause 5
   - Get Message and all of the mandatory aspects of this service as specified in Clause 5
   - Put Message and all of the mandatory aspects of this service as specified in Clause 5.

b) Send and receive XML Instance documents according to the XML Schema specified in Clause 7 in this Technical Specification for the services listed in a).

c) Use the WSDL definitions, SOAP bindings, and operations specified in Clauses 8 and 9.

d) Provide access to the server via HTTPS, and be able to asses that the client digital certificate is valid and that the digital signature as specified in Clause 10 is correct.

5 Service definitions

5.1 List messages

5.1.1 General

The List Messages service is used to obtain a list of available messages for the client according to a given filter (parameters).

The main filter shall be one of the following:

- Application Date of the returned messages
- Server Timestamp of the returned messages
- internal numerical Code of the returned messages

Additional optional filters include:

- Message Identification
- Message Type
- Data Owner

The returned list of messages shall comply with the main filter selected and also with all the optional filters requested, and shall include the following information related to each message:

- Internal numerical code representing the message in the server
- Message Identification
- Message Version
- Status
- Application Time Interval
- Server Timestamp
- Message Type
- Data Owner

5.1.2 Service Request

Parameter Name	Type	M/O/C	Description
StartTime	dateTime	C1	Specifies that the list of messages returned shall only include messages whose end of their Application TimeInterval (Document TimeInterval) or Server Timestamp comes after the provided date.
EndTime	dateTime	C1	Specifies that the list of messages returned shall only include messages whose start of their Application TimeInterval or ServerTimestamp (when the message was received or published in the server) comes before the provided date.
IntervalType	String	C1	Indicates whether the StartTime and EndTime refer to Application TimeInterval or to Server Timestamp. Permitted values: “Application” (default), “Server”.
Code	number	C2	Specifies that the list of messages returned shall only include messages with an internal identification number higher than the provided code. This means that the list will contain messages that are newer to the given one. For optimization purposes, if this filter is used, only messages available since the 00.00 of D-1 (day before) are guaranteed to be included in the response list.
MessageIdentification	string	O	Specifies that the list of messages returned shall only include messages whose Message Identification is compliant with the pattern provided in this parameter. (“*” can be used as a wildcard).
MsgType	string	O	Specifies that the list of messages returned shall only include messages of the provided type.
Owner	string	O	Specifies that the list of messages returned shall only include messages belonging to the provided Owner.

5.1.3 Service Response

If there is no message according to the provided filters, the service shall return an empty list. Otherwise, a list of message descriptors will be returned. Each message descriptor shall include the following parameters:

Parameter Name	Type	M/O/C	Description
Code	Position Type (number)	M	Specifies the internal identification number of the message
MessageIdentification	Identification Type (string)	M	Specifies the Message Identification. Messages defined in IEC 62325 Part 451-X series include this information. For additional messages not included in that standard, the server shall have a way of assigning a MessageIdentification to those messages.
MessageVersion	VersionType	O	Specifies the Message Version. Messages defined in IEC 62325 Part 451-X series include this information. For additional messages not included in that standard, the server may have a way of assigning a Message Version to those messages.
Status	String	O	Specifies the status of messages. Corresponds with the main reason code of the Acknowledgement message associated with this message as per IEC 62325-451-1. Possible values are: OK, FAILED. The status value “OK” corresponds with the IEC 62325-451-1 ReasonCode “A01”, and the status value “FAILED” corresponds with the rest of ReasonCodes.
ApplicationTimeInterval.Start	dateTime	M	Specifies the start of the message Application Time Interval. Messages defined in IEC 62325 Part 451-X series include this information. For additional messages not included in that standard, the server shall have a way of assigning an Application TimeInterval to those messages.
ApplicationTimeInterval.End	dateTime	O	Specifies the end of the message Application Time Interval. Messages defined in IEC 62325 Part 451-X series include this information. For additional messages not included in that standard, the server shall have a way of assigning an Application TimeInterval to those messages. When this information is missing, the message Application Time Interval is “from ApplicationTimeIntervalStart on” without an explicit end.
ServerTimestamp	MessageDateTime Type	M	Specifies the server timestamp (when the message was received or published in the server) of the message.
Type	LongIdentification Type (string)	M	Specifies the Message Type (see Terms and definitions).
Owner	LongIdentification Type (string)	M	Specifies the Data Owner of the message.

5.1.4 Functional requirements

Confidentiality rules of the European energy market for electricity shall be observed, thus the list of messages available to a client will only include those messages to which he/she is entitled (either completely or partially).

A client shall be able to see all his available previously submitted messages to the server, their responses sent from the server (acknowledgements), and any publications that are available to the client.

When the service is called with an invalid filter (e.g. malformed application dates) a Fault message shall be returned.

5.2 Get Message

5.2.1 General

The Get Messages service is used to obtain the message associated to the given parameter (filter).

The filter shall be one of the following:

- Message identification and version
- Message code
- Queue indication

5.2.2 Service Request

Parameter Name	Type	M/O/C	Description
MessageIdentification	Identification Type (string)	C1	Specifies the Message Identification of the requested message.
MessageVersion	VersionType	C1O	Specifies the Message Version of the requested message. If more than one message in the server have the same MessageIdentification and MessageVersion, the most recent one will be returned. If the requested message has no version this parameter is optional.
Code	Position Type (number)	C2	Specifies the internal identification number of the requested message.
Queue	String	C3	Indicates that the server will decide which message will be returned. Its value shall be “NEXT”.

5.2.3 Service Response

Parameter Name	Type	M/O/C	Description
[First child of Payload]	Any[1]	C1	The XML message that is being returned to the client.
BinaryContent	Base64Binary	C2	Optionally binary content may also be returned depending on the type of the requested message.
BinaryName	String	C2	Optionally, the name of the requested binary file.

5.2.4 Functional requirements

Only one message will be retrieved for each Get Message service invocation.

---

[1] Any: any document with any namespace.

A client shall be able to retrieve all his available previously submitted messages to the server, their responses sent from the server (acknowledgements), and any publications that are available to the client.

Servers are entitled to filter parts of the retrieved xml message for confidentiality reasons, when that message, which is available for retrieval, includes information that should not be available to the client.

If the retrieved message is a binary File, then the content is expressed as base 64 encoded wrapped by the tag “BinaryContent”.

When the service is called with an invalid message (e.g. missing or invalid code) a Fault message will be returned.

If a user requests a message to which he/she is not entitled, the system will return a Fault message as if he/she had requested a non-existing message.

The Queue parameter can be used when the server keeps an ordered list of messages for each client to retrieve. A server not supporting this feature will return a fault message.

5.3 Put Message

5.3.1 General

The Put Message service is used to send a message to the server for further processing following the rules of the European energy markets for electricity.

A series of standard XML messages related to the European energy market for electricity are defined in the IEC 62325-451-n series, but this Technical Specification allows servers to process additional XML messages not defined in said series.

Optionally, binary files may also be sent, if supported by the server.

5.3.2 Service Request

Parameter Name	Type	M/O/C	Description
[First child of Payload]	Any[2]	C1	The XML message that is being sent to the server.
BinaryContent	Base64Binary	C2	Optionally binary content may also be sent to the server.
BinaryName	String	C2	Optionally, the name of the binary file sent to the server.

5.3.3 Service Response

The response from the server will be in the form of an XML message indicating the technical and/or functional acceptance or rejection of the message. For the messages described in the IEC 62325-451-n series, the response from the server should be an acknowledgement message as defined in IEC 62325-451-1.

5.3.4 Functional requirements

For each XML message received the server needs to be able to identify each individual message. The IEC 62325-451-n series define such a way via the elements DocumentIdentification and optionally DocumentVersion.

---

2 Any: any document with any namespace.

The server will perform simple validations:

- A user cannot send two (or more) messages with the same identification (e.g. Document Identification and Version). However, another user could send messages with the same identification that other user sent before.
- A user cannot send a message whose Version is lower than another message that has been sent previously with the same Identification by the same user.

If there is not enough information in the sent message to create a proper response (incomplete XML or missing tags), a Fault message will be issued instead.

If a user is not authorized to send a specific message, a Fault message shall be returned.

5.4 Query Data

5.4.1 General

The non-mandatory Query Data Service can be used by clients to request specific data from the server using different query parameters.

The server does not need to have the response XML message ready before the service invocation, and can create a specific message in response to each request as needed.

5.4.2 Service Request

Parameter Name	Type	M/O	Description
DataType	String	M	Indicates the type of data being requested.
StartTime	dateTime	O	Specifies that the returned message shall only include data whose Application Date is after the provided date.
EndTime	dateTime	O	Specifies that the returned message shall only include data whose Application Date is before the provided date.
Any[3]	String	O	Specifies additional parameters for the service.

5.4.3 Service Response

Parameter Name	Type	M/O/C	Description
QueryData	gop:QueryData	M	Wraps the XML message that is being returned to the client and the parameters used in the request.

5.4.4 Functional requirements

Only one message will be retrieved for each Query Data service invocation.

When the service is called with invalid parameters (e.g. malformed application dates) a Fault message will be returned.

A list of basic datatypes to be supported if this service is implemented is shown below:

[3] Any: Any other parameter indicated with the pairs “name” and “value” with the element Option in the message header.

DataType value	M/O	Description
“listOfDataTypes”	M	The server will return a list of valid DataTypes that can be used for this service on this server.
“serverTimestamp”	M	The server will return the Server Timestamp in UTC time.
“parameterLimits”	O	The server will return its operational limits for parameters used in the List, Get, Put services, such as maximum message size, maximum number of queried days in the list service, etc. If there is no a specific limit for a given parameter, the response will not include such limit value. If the user breaches one of the specified limits, the server will return a Fault message instead of the Response of the operation used. The following non comprehensive list of parameter names should be used to indicate such limits: • MaxNumMessagesInListResponse: Maximum number of messages that will be returned in a list operation response. • NumberOfDaysForLowCodeInListResponse: Number of days that are guarantee to be included in the response list when the request has used a small code value (typically 0). According to this specification, the default value for this parameter limit is 1 (all messages available from 00:00 of D-1). • MaxApplicationTimeIntervalInDaysInListRequest: Maximum number of days that a request for Application time interval type can span. • MaxServerTimeIntervalInDaysInListRequest: Maximum number of days that a request for Server time interval type can span. • MaxPayloadSizeInMBInPutRequest: Maximum size, in Megabytes, that message payload content can have. Messages with bigger size will be rejected. • MaxGetRequestPerMinute: Number of Get operations per minute that a user can execute. • MaxPutRequestPerMinute: Number of Put operations per minute that a user can execute. • MaxListRequestPerMinute: Number of List operations per minute that a user can execute. • MaxQueryRequestPerMinute: Number of Query Data operations per minute that a user can execute. • MaxMessageAgeInDays: Max number of days that a message will be accessible by this Technical Specification operations. • MaxDiffServerTimestampInSeconds: If set, the server will reject messages that do not meet the following criteria: $CT - ST + MD >= 0$ Being CT the current server time, ST the msg:serverTimestamp indicated in the request message and MD this parameter value. Other parameters may be added.

In order to ensure non-repudiation, the service response will include the parameters used in the invocation along with the response.

6 Applying IEC 61968-100

6.1 Integration Pattern

6.1.1 General

The interactions between Client and Server described in this Technical Specification correspond with the IEC 61968-100 use case “Simple Request/Reply” supported by the “Basic Web Service Integration Pattern”.

This does not preclude the implementation of other Integration Patterns to support additional use cases such us “Request/Reply with ESB”, “Events” etc.

The server will expose in a WSDL a single operation “request” that will serve as the single entry point for the services described in this Technical Specification.

In Figures 1 to 3 the request/reply exchange between client and server is shown, including the type of message sent, the IEC 61968-100 verb and noun used, and the payload when applicable.

6.1.2 List Service

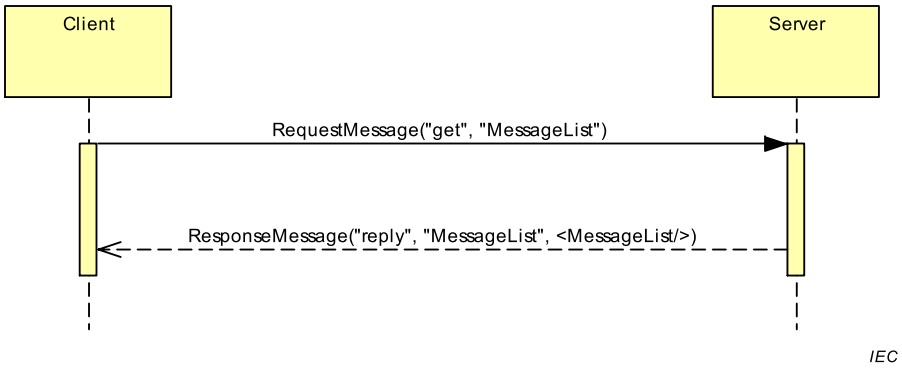

Figure 1 – List Service Sequence Diagram

6.1.3 Get Service

In this example, a message of type “Publication_MarketDocument” is requested:

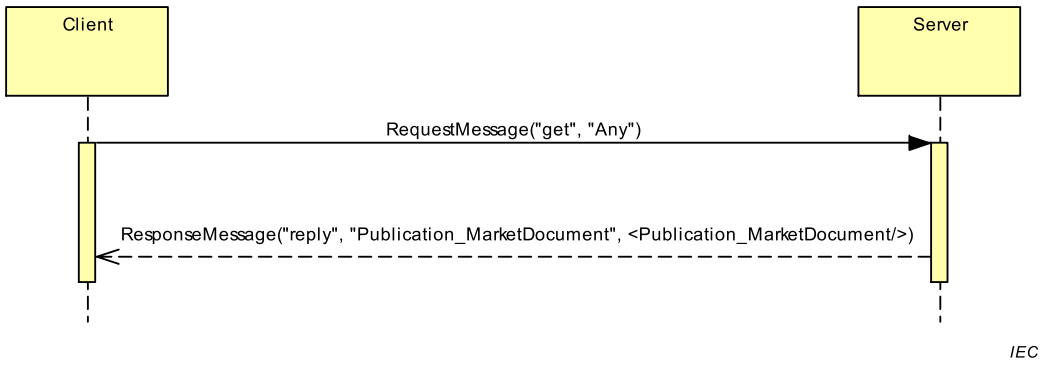

Figure 2 – Get Service Sequence Diagram

6.1.4 Put Service

In this example, a message of type “Publication_MarketDocument” is sent to the server:

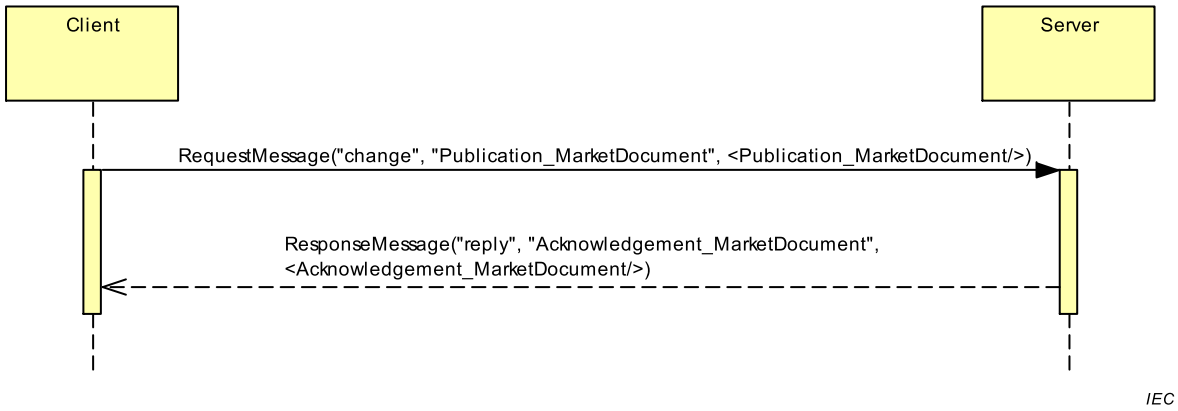

Figure 3 – Put Service Sequence Diagram

6.2 Service mapping

6.2.1 General

The mapping between the different services described in this Technical Specification and the IEC 61968-100 messages, including some relevant parameters, is shown in the following table:

Service	IEC 61968-100 Stereotype Msg.	IEC 61968-100 Verb	IEC 61968-100 Noun	IEC 61968-100 Payload
List (Req)	RequestMessage	get	MessageList	n/a
List (Resp)	ResponseMessage	reply	MessageList	gop:MessageList
Get (Req)	RequestMessage	get	Any[4]	n/a
Get (Resp)	ResponseMessage	reply	*ChildOfPayload*	*xsd:any*[5]
Put (Req)	RequestMessage	create	*ChildOfPayload*	*xsd:any*[5]
Put (Resp)	ResponseMessage	reply	*ChildOfPayload*	*xsd:any*[5]
QueryData (Req)	RequestMessage	get	QueryData	n/a
QueryData (Resp)	ResponseMessage	reply	QueryData	gop:QueryData

Unless otherwise stated, all elements in the Header, Request or Response defined in IEC 61968-100 can be used.

6.2.2 Header values

The “Source” element, if used, will be the EIC code of the Sender Entity for messages following the IEC 62325-451-n series.

It is strongly recommended to use the element “Timestamp” for all the interchanges. This gives an added value to the signature allowing the recipient to reject messages with timestamp set in the future. The QueryData “serverTimestamp” response uses this element to indicate the required value.

---

4 Any: Literal text “Any”.

5 *xsd:any*: any document with any namespace.

6.2.3 Request values

The parameters in the List Service Request, Get Service Request and Query Service Request that do not correspond directly with a parameter in the IEC 61968-100 msg:RequestMessage will be provided as Name/Value pairs under the msg:Option element of the msg:Request.

In the case of the Put service Request, when sending a binary file, the name of the file will be included in the msg:Request.ID, with the attribute idType equal to “name”.

6.2.4 Response values

In the case of the Put service Response, the msg:Reply.Result values “OK” and “FAILED” correspond with the “Completely Accepted” and “Completely Rejected” responses described in IEC 62325-451-1.

In the case of the Get service Response, when requesting a binary file, the name of the file will be included in the msg:Response.ID, with the attribute idType equal to “name”.

In the case of the Query service Response, all the attributes (filters) that were included in the msg:Option element in the request will be included in the QueryData response.

6.2.5 Payload values

The generic message payload container will be used (it is not mandatory to have strongly typed WSDLs)

The “BinaryContent” described in this Technical Specification will be mapped to the “msg:Compressed” element of IEC 61968-100.

The “msg:Format” element will be absent or have the value “XML” if the payload is an XML, and “BINARY” if the payload is a binary document.

7 Schema definitions

7.1 Common definitions

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="urn:iec62325.504:messages:1:0"
targetNamespace="urn:iec62325.504:messages:1:0"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:annotation>
    <xsd:documentation>TF EDI EME – Common Types</xsd:documentation>
  </xsd:annotation>
  <xsd:complexType name="TimeIntervalType">
    <xsd:sequence>
      <xsd:element name="start" type="xsd:dateTime"/>
      <xsd:element name="end" type="xsd:dateTime" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:simpleType name="StatusType">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="OK"/>
      <xsd:enumeration value="FAILED"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:element name="Parameter">
    <xsd:complexType>
```

```
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="value" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

7.2 List message

The graphical representation of the following schema can be seen in Figure 4:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="urn:iec62325.504:messages:1:0"
targetNamespace="urn:iec62325.504:messages:1:0"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xsd:annotation>
      <xsd:documentation>TF EDI EME - List</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="MessageList">
      <xsd:annotation>
         <xsd:documentation>List of messages</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Message" minOccurs="0"
maxOccurs="unbounded">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="Code" type="xsd:positiveInteger"/>
                     <xsd:element name="MessageIdentification"
type="xsd:string"/>
                     <xsd:element name="MessageVersion"
type="xsd:positiveInteger" minOccurs="0"/>
                     <xsd:element name="Status" type="StatusType"
minOccurs="0"/>
                     <xsd:element name="ApplicationTimeInterval"
type="TimeIntervalType"/>
                     <xsd:element name="ServerTimestamp"
type="xsd:dateTime"/>
                     <xsd:element name="Type" type="xsd:string"/>
                     <xsd:element name="Owner" type="xsd:string"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

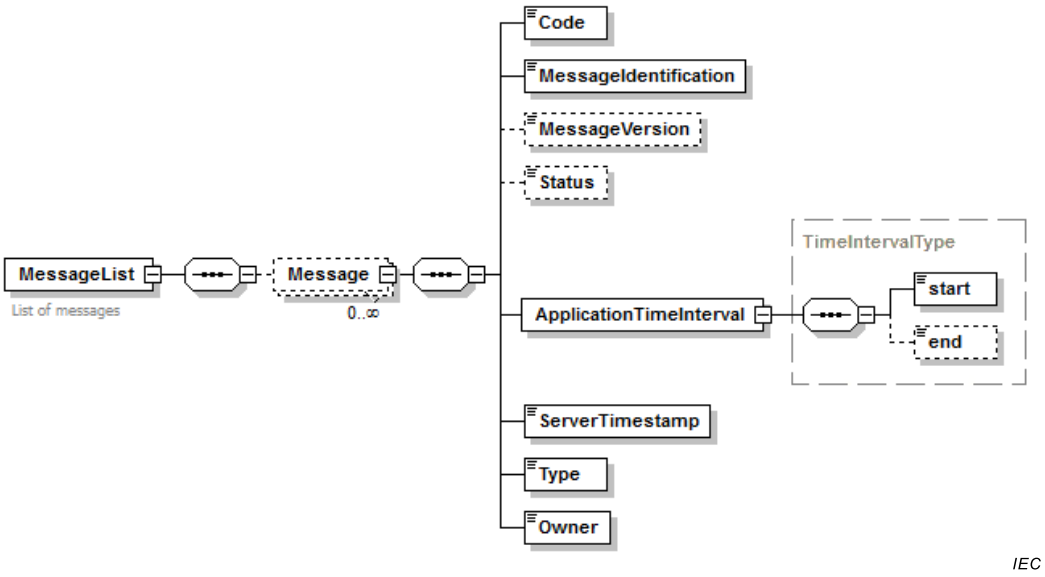

Figure 4 – MessageList schema structure

7.3 QueryData message

The graphical representation of the following schema can be seen in Figure 5:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="urn:iec62325.504:messages:1:0"
targetNamespace="urn:iec62325.504:messages:1:0"
elementFormDefault="qualified" attributeFormDefault="unqualified">
<xsd:annotation>
     <xsd:documentation>TF EDI EME – QueryData</xsd:documentation>
  </xsd:annotation>
  <xsd:element name="QueryData">
     <xsd:annotation>
       <xsd:documentation>Wraps the response and request parameters from
the "QueryData" service</xsd:documentation>
     </xsd:annotation>
     <xsd:complexType>
       <xsd:sequence>
         <xsd:element name="RequestParameters">
           <xsd:complexType>
             <xsd:sequence>
                <xsd:element ref="Parameter" maxOccurs="unbounded"/>
             </xsd:sequence>
           </xsd:complexType>
         </xsd:element>
         <xsd:any namespace="##any" processContents="lax" minOccurs="0"/>
       </xsd:sequence>
     </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

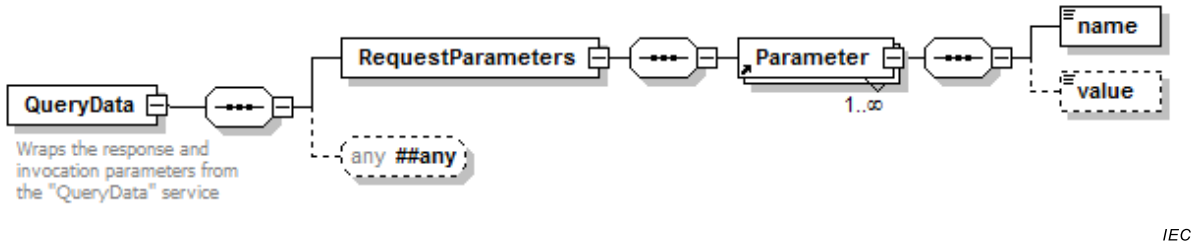

Figure 5 – QueryData schema structure

7.4 QueryData List of data types

The graphical representation of the following schema can be seen in Figure 6:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="urn:iec62325.504:messages:1:0"
targetNamespace="urn:iec62325.504:messages:1:0"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xsd:annotation>
     <xsd:documentation>TF EDI EME – QueryData List of
Parameters</xsd:documentation>
  </xsd:annotation>
  <xsd:element name="ParameterList">
     <xsd:annotation>
       <xsd:documentation>Parameter list with name and optional
value</xsd:documentation>
     </xsd:annotation>
     <xsd:complexType>
       <xsd:sequence maxOccurs="unbounded">
         <xsd:element ref="Parameter"/>
       </xsd:sequence>
     </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

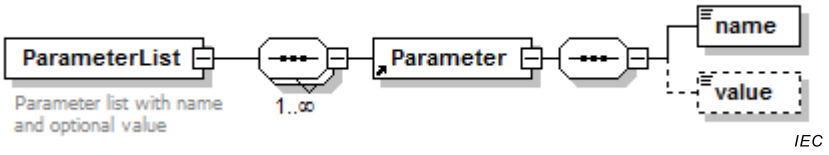

Figure 6 – ParameterList schema structure

8 Service Provider WSDL abstract definitions

By using synchronous web services, a client is immediately aware of the result of the service invocation.

This clause specifies the abstract WSDL definitions to support the services specified in this Technical Specification.

```
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
```

```
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:cmsg="urn:iec62325.504:messages:1:0"
xmlns:wss="urn:iec62325.504:wss:1:0"
xmlns:msg="http://iec.ch/TC57/2011/schema/message"
     targetNamespace="urn:iec62325.504:wss:1:0">
      <types>
            <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
                  <xs:import namespace="urn:iec62325.504:messages:1:0"
schemaLocation="./urn-iec62325-504-messages-1-0.xsd"/>
                  <xs:import
namespace="http://iec.ch/TC57/2011/schema/message" schemaLocation="./http-
iec-ch-TC57-2011-schema-message.xsd"/>
            </xs:schema>
      </types>
      <message name="msgRequestMessage">
            <part name="parameter" element="msg:RequestMessage"/>
      </message>
      <message name="msgResponseMessage">
            <part name="parameter" element="msg:ResponseMessage"/>
      </message>
      <message name="msgFaultMsg">
            <part name="msgFaultMessage" element="msg:FaultMessage"/>
      </message>
      <portType name="port_TFEDI_type">
            <operation name="request">
                  <input message="wss:msgRequestMessage"/>
                  <output message="wss:msgResponseMessage"/>
                  <fault name="msgFaultMessage" message="wss:msgFaultMsg"/>
            </operation>
      </portType>
</definitions>
```

9 Service Provider WSDL SOAP binding

This clause specifies the binding template of the abstract WSDL definitions in Clause 8 with SOAP Messages with the http(s) transport protocol. Although SOAP 1.2 is shown here, SOAP 1.1 may also be used.

The graphical representation of the following definitiions can be seen in Figure 7:

```
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:cmsg="urn:iec62325.504:messages:1:0"
xmlns:wss="urn:iec62325.504:wss:1:0"
xmlns:msg="http://iec.ch/TC57/2011/schema/message"
targetNamespace="urn:iec62325.504:wss:1:0">
  <binding name="binding_TFEDI" type="wss:port_TFEDI_type">
    <soap12:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
     <operation name="request">
       <soap12:operation soapActionRequired="false" style="document"/>
       <input>
         <soap12:body use="literal"/>
```

```
        </input>
        <output>
          <soap12:body use="literal"/>
        </output>
        <fault name="msgFaultMessage">
          <soap12:fault name="msgFaultMessage" use="literal"/>
        </fault>
      </operation>
    </binding>
    <service name="ServiceEME">
      <port name="Service_EME_Port" binding="wss:binding_TFEDI">
        <soap12:address
location="https://example.com/WebService_EME/Service_EME"/>
      </port>
    </service>
</definitions>
```

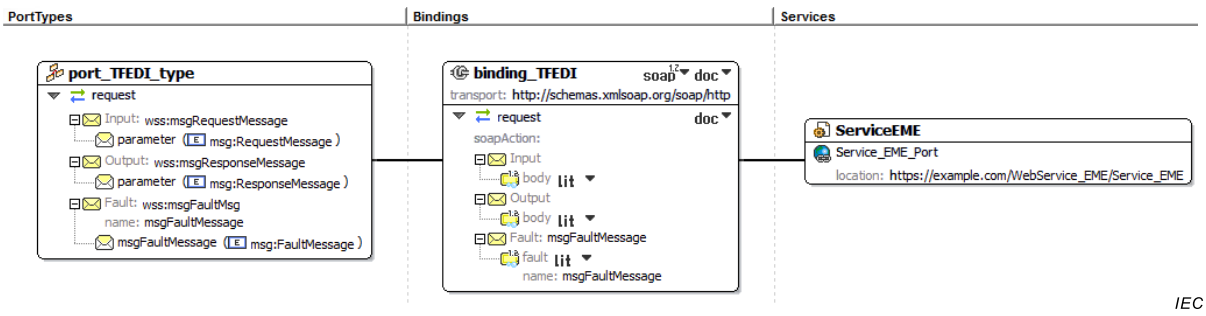

Figure 7 – WSDL structure

Fault messages are used when the application produces an error condition that cannot be expressed with the operation response. Those errors are not usually related to business rules.

The fault message consists of two parts, a code and reason (also known as a fault string) and, optionally a detailed block that can be used to include more information about the error.

10 Security

The confidentiality of communications, and therefore the confidentiality of the information transmitted are achieved in the transport layer using https with client and server certificates (2-way SSL).

All the involved certificates shall conform with the X.509 v3 specification (see RFC 5280 and RFC 6818).

Since this Technical Specification relies on HTTPS in order to cipher and authenticate communications, and as such, also relies on Transport Layer Security (TLS, see RFC 4346) for security services, special care must be taken when securing the transport layer. Thus, in order to conform to this Technical Specification, TLS clients and servers shall use TLS1.1 or higher (see RFC 6176). Negotiating the use of SSL3 and/or TLS1.0 is not recommended, and its use should be explicitly agreed upon by the involved parties.

To ensure the authentication of the signer, the document integrity and non-repudiation, all IEC 61968-100 messages with XML documents in Payloads for the “Get”, “Put” and “QueryData” services are digitally signed either by the Data Owner or the Data Provider.

This digital signature, made with the private key associated with the user's digital certificate, follows the "XML Digital Signature" standard, and is transmitted in the signed message (enveloped signature). The signature is included in the IEC 61968-100 messages Header.

The signature applies to the complete IEC 61968-100 message that is being sent or retrieved, not to the XML Soap, nor exclusively to the single XML document in the Payload.

Thus the Reference URI shall always be “” in order to point to the whole IEC 61968-100 document.

The CanonicalizationMethod will be one of:

- http://www.w3.org/TR/2001/REC-xml-c14n-20010315
- http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments
- http://www.w3.org/2001/10/xml-exc-c14n
- http://www.w3.org/2001/10/xml-exc-c14n#WithComments

The set of supported transforms (in this order) are:

- http://www.w3.org/2000/09/xmldsig#enveloped-signature
- (optional) any of the permitted canonicalization algorithms

In particular, no XPath or XSLT transforms are allowed.

The element “KeyInfo” includes the public key of the certificate used to create the signature.

In the case of messages with binary content, document signing is similar; since the “Compressed” element falls inside the IEC 61968-100 message that is being signed.

Annex A
(normative)

XML schema for common IEC 62325-504 messages

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Common messages for IEC 62325-504 operations     -->
<!--       Change Log                                  -->
<!-- 2014/09/12  Update after CD review                -->
<!-- 2013/06/12  First version                         -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="urn:iec62325.504:messages:1:0"
targetNamespace="urn:iec62325.504:messages:1:0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <!--_____________________________________________________-->
   <xsd:annotation>
      <xsd:documentation>TF EDI EME - Common Types</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType name="TimeIntervalType">
      <xsd:sequence>
         <xsd:element name="start" type="xsd:dateTime"/>
         <xsd:element name="end" type="xsd:dateTime" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="StatusType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="OK"/>
         <xsd:enumeration value="FAILED"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="Parameter">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="name" type="xsd:string"/>
            <xsd:element name="value" type="xsd:string" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <!--_____________________________________________________-->
   <xsd:annotation>
      <xsd:documentation>TF EDI EME - List</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="MessageList">
      <xsd:annotation>
         <xsd:documentation>List of messages</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Message" minOccurs="0" maxOccurs="unbounded">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="Code" type="xsd:positiveInteger"/>
                     <xsd:element name="MessageIdentification" type="xsd:string"/>
                     <xsd:element name="MessageVersion" type="xsd:positiveInteger"
minOccurs="0"/>
                     <xsd:element name="Status" type="StatusType" minOccurs="0"/>
                     <xsd:element name="ApplicationTimeInterval"
type="TimeIntervalType"/>
```

```
                  <xsd:element name="ServerTimestamp" type="xsd:dateTime"/>
                  <xsd:element name="Type" type="xsd:string"/>
                  <xsd:element name="Owner" type="xsd:string"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<!--_____________________________________________________-->
<xsd:annotation>
   <xsd:documentation>TF EDI EME - QueryData</xsd:documentation>
</xsd:annotation>
<xsd:element name="QueryData">
   <xsd:annotation>
      <xsd:documentation>Wraps the response and request parameters from the
"QueryData" service</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="RequestParameters">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element ref="Parameter" maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:any namespace="##any" processContents="lax" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<!--_____________________________________________________-->
<xsd:annotation>
   <xsd:documentation>TF EDI EME - QueryData List of
Parameters</xsd:documentation>
</xsd:annotation>
<xsd:element name="ParameterList">
   <xsd:annotation>
      <xsd:documentation>Parameter list with name and optional
value</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
      <xsd:sequence maxOccurs="unbounded">
         <xsd:element ref="Parameter"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
</xsd:schema>
```

Annex B
(informative)

Message examples

B.1 List

B.1.1 Basic example – Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
      <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
            <msg:Verb>get</msg:Verb>
            <msg:Noun>MessageList</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
            <msg:Timestamp>2012-11-30T09:30:47.581Z</msg:Timestamp>
            <msg:AckRequired>true</msg:AckRequired>
         </msg:Header>
         <msg:Request>
            <msg:StartTime>2012-11-26T23:00:00Z</msg:StartTime>
            <msg:EndTime>2012-11-27T23:00:00Z</msg:EndTime>
            <msg:Option>
               <msg:name>IntervalType</msg:name>
               <msg:value>Server</msg:value>
            </msg:Option>
            <msg:Option>
               <msg:name>Owner</msg:name>
               <msg:value>10XEXAMPLE-EIC-P</msg:value>
            </msg:Option>
         </msg:Request>
      </msg:RequestMessage>
   </soap:Body>
</soap:Envelope>
```

B.1.2 Basic example – Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
  <soap:Header/>
  <soap:Body>
   <msg:ResponseMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
     <msg:Header>
     <msg:Verb>reply</msg:Verb>
     <msg:Noun>MessageList</msg:Noun>
     <msg:Context>PRODUCTION</msg:Context>
     <msg:Timestamp>2014-11-30T09:30:48.321Z</msg:Timestamp>
   </msg:Header>
   <msg:Reply>
     <msg:Result>OK</msg:Result>
   </msg:Reply>
   <msg:Payload>
     <urn:MessageList xmlns:urn="urn:iec62325.504:messages:1:0">
       <urn:Message>
         <urn:Code>6465</urn:Code>
    <urn:MessageIdentification>NTCForecast_20140917</urn:MessageIdentification>
         <urn:MessageVersion>1</urn:MessageVersion>
         <urn:Status>FAILED</urn:Status>
```

```
          <urn:ApplicationTimeInterval>
            <urn:start>2014-09-16T22:00:00Z</urn:start>
          </urn:ApplicationTimeInterval>
          <urn:ServerTimestamp>2014-09-16T09:00:06Z</urn:ServerTimestamp>
          <urn:Type>Publication_MarketDocument</urn:Type>
          <urn:Owner>10XEXAMPLE-EIC-P</urn:Owner>
       </urn:Message>
       <urn:Message>
          <urn:Code>6466</urn:Code>
  <urn:MessageIdentification>AckNTCForecast_20140917</urn:MessageIdentificati
on>
          <urn:MessageVersion>1</urn:MessageVersion>
          <urn:Status>FAILED</urn:Status>
          <urn:ApplicationTimeInterval>
            <urn:start>2014-09-16T22:00:00Z</urn:start>
          </urn:ApplicationTimeInterval>
          <urn:ServerTimestamp>2014-09-16T09:00:06Z</urn:ServerTimestamp>
          <urn:Type>Acknowledgement_MarketDocument</urn:Type>
          <urn:Owner>10XEXAMPLE-EIC-P</urn:Owner>
       </urn:Message>
       <urn:Message>
          <urn:Code>6467</urn:Code>
  <urn:MessageIdentification>PhysicalFlows_2014091611</urn:MessageIdentificat
ion>
          <urn:MessageVersion>1</urn:MessageVersion>
          <urn:Status>OK</urn:Status>
          <urn:ApplicationTimeInterval>
            <urn:start>2014-09-16T08:00:00Z</urn:start>
            <urn:end>2014-09-16T09:00:00Z</urn:end>
          </urn:ApplicationTimeInterval>
          <urn:ServerTimestamp>2014-09-16T09:13:04Z</urn:ServerTimestamp>
          <urn:Type>Publication_MarketDocument</urn:Type>
          <urn:Owner>10XEXAMPLE-EIC-P</urn:Owner>
       </urn:Message>
       <urn:Message>
          <urn:Code>6468</urn:Code>
  <urn:MessageIdentification>AckPhysicalFlows_2014091611</urn:MessageIdentifi
cation>
          <urn:MessageVersion>1</urn:MessageVersion>
          <urn:Status>OK</urn:Status>
          <urn:ApplicationTimeInterval>
            <urn:start>2014-09-16T08:00:00Z</urn:start>
            <urn:end>2014-09-16T09:00:00Z</urn:end>
          </urn:ApplicationTimeInterval>
          <urn:ServerTimestamp>2014-09-16T09:13:04Z</urn:ServerTimestamp>
          <urn:Type>Acknowledgement_MarketDocument</urn:Type>
          <urn:Owner>10XEXAMPLE-EIC-P</urn:Owner>
       </urn:Message>
     </urn:MessageList>
   </msg:Payload>
 </msg:ResponseMessage>
 </soap:Body>
</soap:Envelope>
```

B.2 Get

B.2.1 General

As a general rule, before retrieving a Message, the client should find out if the message exists.

Thus, the recommended sequence diagram is shown in Figure B.1, where the actor requests the ”List” service in first place, recovering the list of available messages. Once the actor has such information, he will invoke the “Get” service for a message that appears in the retrieved list.

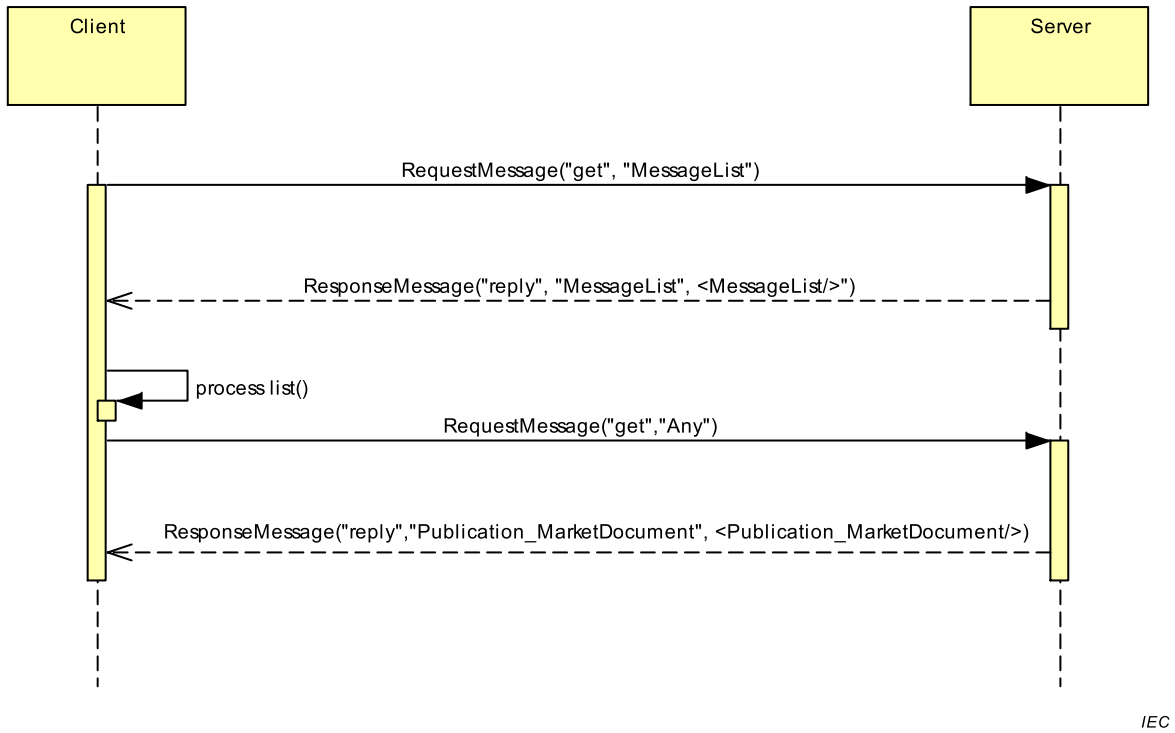

Figure B.1 – List and Get Sequence Diagram

B.2.2 Basic example

B.2.2.1 Request by code:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
    <soap:Body>
       <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
            <msg:Header>
                <msg:Verb>get</msg:Verb>
                <msg:Noun>Any</msg:Noun>
                <msg:Context>PRODUCTION</msg:Context>
                <msg:Timestamp>2012-11-30T09:30:47.581Z</msg:Timestamp>
                <msg:AckRequired>true</msg:AckRequired>
            </msg:Header>
            <msg:Request>
                <msg:Option>
                    <msg:name>Code</msg:name>
                    <msg:value>879021</msg:value>
                </msg:Option>
            </msg:Request>
```

```
        </msg:RequestMessage>
    </soap:Body>
</soap:Envelope>
```

B.2.2.2 Request by queue:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
    <soap:Body>
       <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
            <msg:Header>
                <msg:Verb>get</msg:Verb>
                <msg:Noun>Any</msg:Noun>
                <msg:Context>PRODUCTION</msg:Context>
                <msg:Timestamp>2012-11-30T09:30:47.581Z</msg:Timestamp>
                <msg:AckRequired>true</msg:AckRequired>
            </msg:Header>
            <msg:Request>
                <msg:Option>
                    <msg:name>Queue</msg:name>
                    <msg:value>NEXT</msg:value>
                </msg:Option>
            </msg:Request>
       </msg:RequestMessage>
    </soap:Body>
</soap:Envelope>
```

B.2.2.3 Request by identification and version:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
    <soap:Body>
       <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
            <msg:Header>
                <msg:Verb>get</msg:Verb>
                <msg:Noun>Any</msg:Noun>
                <msg:Context>PRODUCTION</msg:Context>
                <msg:Timestamp>2012-11-30T09:30:47.581Z</msg:Timestamp>
                <msg:AckRequired>true</msg:AckRequired>
            </msg:Header>
            <msg:Request>
                <msg:Option>
                    <msg:name>MessageIdentification</msg:name>
                    <msg:value>Schedule_D_20140416</msg:value>
                </msg:Option>
                <msg:Option>
                    <msg:name>MessageVersion</msg:name>
                    <msg:value>1</msg:value>
                </msg:Option>
            </msg:Request>
       </msg:RequestMessage>
    </soap:Body>
</soap:Envelope>
```

B.2.2.4 Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
```

```
   <soap:Header>
   </soap:Header>
   <soap:Body>
       <msg:ResponseMessage
xmlns:msg="http://iec.ch/TC57/2011/schema/message">
            <msg:Header>
                <msg:Verb>reply</msg:Verb>
                <msg:Noun>Schedule_MarketDocument</msg:Noun>
                <msg:Context>PRODUCTION</msg:Context>
                <msg:Timestamp>2012-11-30T09:30:48.397Z</msg:Timestamp>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
         <SignedInfo>
            <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-
xml-c14n-20010315"/>
            <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"/>
            <Reference URI="">
               <Transforms>
                  <Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                  <Transform Algorithm="http://www.w3.org/TR/2001/REC-xml-
c14n-20010315"/>
               </Transforms>
               <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
               <DigestValue>xSnE7qGXJkWGG11rq3ze5DYGxHw=</DigestValue>
            </Reference>
         </SignedInfo>
<SignatureValue>QLk/r49g4uoR8pXT9WX3fg6QYEh/r0……==</SignatureValue>
         <KeyInfo>
            <X509Data>
               <X509IssuerSerial>
                  <X509IssuerName>CN=testCA, O=test Domain</X509IssuerName>
                  <X509SerialNumber>238</X509SerialNumber>
               </X509IssuerSerial>
               <X509SubjectName>UID=myname, OU=intern, OU=myou, OU=users,
O=test, C=test</X509SubjectName>
               <X509Certificate>MIID2TCCAsGgAwIBA ....  =</X509Certificate>
            </X509Data>
         </KeyInfo>
      </Signature>
            </msg:Header>
            <msg:Reply>
                <msg:Result>OK</msg:Result>
            </msg:Reply>
            <msg:Payload>
<Schedule_MarketDocument xmlns="urn:iec62325.351:tc57wg16:451-
2:scheduledocument:5:0">
   <mRID>Schedule_D_20140416</mRID>
   <revisionNumber>1</revisionNumber>
   <type>A04</type>
   <process.processType>A01</process.processType>
   <process.classificationType>A01</process.classificationType>
   <sender_MarketParticipant.mRID codingScheme="A01">10XEXAMPLE-EIC-P
</sender_MarketParticipant.mRID>
   <sender_MarketParticipant.marketRole.type>A04
</sender_MarketParticipant.marketRole.type>
   <receiver_MarketParticipant.mRID codingScheme="A01">10XEXAMPLE-EIC-P
</receiver_MarketParticipant.mRID>
```

```
<receiver_MarketParticipant.marketRole.type>A04</receiver_MarketParticipant.ma
rketRole.type>
   <createdDateTime>2014-04-15T13:06:29Z</createdDateTime>
   <schedule_Time_Period.timeInterval>
      <start>2014-04-15T22:00Z</start>
      <end>2014-04-16T22:00Z</end>
   </schedule_Time_Period.timeInterval>
...
</Schedule_MarketDocument>
            </msg:Payload>
        </msg:ResponseMessage>
      </soap:Body>
</soap:Envelope>
```

B.3 Put

B.3.1 Basic example

B.3.1.1 Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
    <soap:Body>
       <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
           <msg:Header>
               <msg:Verb>create</msg:Verb>
               <msg:Noun>Schedule_MarketDocument</msg:Noun>
               <msg:Context>PRODUCTION</msg:Context>
               <msg:Timestamp>2014-04-15T13:06:29.885Z</msg:Timestamp>
               <msg:AckRequired>true</msg:AckRequired>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
     ...
     </Signature>
         </msg:Header>
         <msg:Payload>
<Schedule_MarketDocument xmlns="urn:iec62325.351:tc57wg16:451-
2:scheduledocument:5:0">
   <mRID>Schedule_D_20140416</mRID>
   <revisionNumber>1</revisionNumber>
   <type>A04</type>
   <process.processType>A01</process.processType>
   <process.classificationType>A01</process.classificationType>
   <sender_MarketParticipant.mRID codingScheme="A01">10XEXAMPLE-EIC-P
</sender_MarketParticipant.mRID>
   <sender_MarketParticipant.marketRole.type>A04
</sender_MarketParticipant.marketRole.type>
   <receiver_MarketParticipant.mRID codingScheme="A01">10XEXAMPLE-EIC-P
</receiver_MarketParticipant.mRID>
   <receiver_MarketParticipant.marketRole.type>A04
</receiver_MarketParticipant.marketRole.type>
   <createdDateTime>2014-04-15T13:06:29Z</createdDateTime>
...
</Schedule_MarketDocument>
         </msg:Payload>
      </msg:RequestMessage>
   </soap:Body>
</soap:Envelope>
```

B.3.1.2 Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
    <soap:Body>
      <msg:ResponseMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
            <msg:Header>
                <msg:Verb>reply</msg:Verb>
                <msg:Noun>Acknowledgement_MarketDocument</msg:Noun>
                <msg:Context>PRODUCTION</msg:Context>
                <msg:Timestamp>2014-04-15T13:07:31.380Z</msg:Timestamp>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      ...
      </Signature>
            </msg:Header>
            <msg:Reply>
                <msg:Result>OK</msg:Result>
            </msg:Reply>
            <msg:Payload>
<Acknowledgement_MarketDocument xmlns="urn:iec62325.351:tc57wg16:451-
1:acknowledgementdocument:6:0">
   <mRID>ACK_Schedule_D_20140416</mRID>
   <createdDateTime>2014-04-15T13:07:30Z</createdDateTime>
   <sender_MarketParticipant.mRID codingScheme="A01">10XEXAMPLE-EIC-P
</sender_MarketParticipant.mRID>
   <sender_MarketParticipant.marketRole.type>A04
</sender_MarketParticipant.marketRole.type>
   <receiver_MarketParticipant.mRID codingScheme="A01">10XEXAMPLE-EIC-P
</receiver_MarketParticipant.mRID>
   <receiver_MarketParticipant.marketRole.type>A04
</receiver_MarketParticipant.marketRole.type>
   <received_MarketDocument.mRID>Schedule_D_20140416
</received_MarketDocument.mRID>
   <received_MarketDocument.revisionNumber>1
</received_MarketDocument.revisionNumber>
   <received_MarketDocument.type>A51</received_MarketDocument.type>
   <received_MarketDocument.createdDateTime>2014-04-15T13:06:29Z
</received_MarketDocument.createdDateTime>
   <Reason>
      <code>A01</code>
   </Reason>
</Acknowledgement_MarketDocument>
            </msg:Payload>
         </msg:ResponseMessage>
    </soap:Body>
</soap:Envelope>
```

B.3.2 Example with binary data

If the message has a binary part, this may be transmitted as 64b encoded form:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
       <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
            <msg:Header>
                <msg:Verb>create</msg:Verb>
                <msg:Noun>Compressed</msg:Noun>
                <msg:Context>PRODUCTION</msg:Context>
                <msg:Timestamp>2012-11-30T09:33:47.658Z</msg:Timestamp>
                <msg:AckRequired>true</msg:AckRequired>
```

```
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      ...
      </Signature>
            </msg:Header>
            <msg:Request>
               <msg:ID idType="name">schedule_xyz.jpg</msg:ID>
            </msg:Request>
            <msg:Payload>
                <msg:Compressed>
/9j/4AAQSkZJRgABAgEASABIAAD/wAALCAAGABcBAREA/9sAhAAGBAUGBQQGBgUGBwcGCAoRCwoJ
CQoVDxAMERkWGhoYFhgXGx8oIRsdJR4XGCIvIyUpKiwtLBshMTQwKzQoKywrAQcHBwoJChQLCxQr
HBgcHCsrKysrKysrKysrKysrKysrKysrKysrKysrKysrKysrKysrKysrKysrKysrKysrKyv/xADS
AAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgsQAAIBAwMCBAMFBQQEAAABfQECAwAEEQUSITFB
BhNRYQcicRQygZGhCCNCscEVUtHwJDNicoIJChYXGBkaJSYnKCkqNDU2Nzg5OkNERUZHSElKU1RV
VldYWVpjZGVmZ2hpanN0dXZ3eHl6g4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrC
w8TFxsfIycrS09TV1tfY2drh4uPk5ebn6Onq8fLz9PX29/j5+v/aAAgBAQAAPwDtfA+l6zHpXgv+
y9euZdSu/A11/Z/22OHyrKUrY7NuyIEqGK537zhR15zx8vidPFXwakvIb++vYIdQ1OCM3kLRvEBo
10xUF55ncZYtueQn5iAAqqK0Pixffb9F+JGla3qupR2+meX/AGs8SfaFPnXcMlh9niZ1VdkAZJMG
PLNk+YRur//Z</msg:Compressed>
                <msg:Format>BINARY</msg:Format>
            </msg:Payload>
       </msg:RequestMessage>
    </soap:Body>
</soap:Envelope>
```

B.4 Query Data

B.4.1 List of data types example

B.4.1.1 Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
      <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
            <msg:Verb>get</msg:Verb>
            <msg:Noun>QueryData</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
            <msg:Timestamp>2012-11-30T09:30:47.581Z</msg:Timestamp>
         </msg:Header>
         <msg:Request>
            <msg:Option>
               <msg:name>DataType</msg:name>
               <msg:value>listOfDataTypes</msg:value>
            </msg:Option>
         </msg:Request>
      </msg:RequestMessage>
   </soap:Body>
</soap:Envelope>
```

B.4.1.2 Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
      <msg:ResponseMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
            <msg:Verb>reply</msg:Verb>
            <msg:Noun>QueryData</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
```

```
            <msg:Timestamp>2012-11-30T09:30:48.004Z</msg:Timestamp>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      ...
      </Signature>
         </msg:Header>
         <msg:Reply>
            <msg:Result>OK</msg:Result>
         </msg:Reply>
         <msg:Payload>
            <QueryData xmlns="urn:iec62325.504:messages:1:0">
               <RequestParameters>
                  <Parameter>
                     <name>DataType</name>
                     <value>listOfDataTypes</value>
                  </Parameter>
               </RequestParameters>
               <ParameterList xmlns="urn:iec62325.504:messages:1:0">
                     <Parameter>
                       <name>listOfDataTypes</name>
                     </Parameter>
                     <Parameter>
                       <name>serverTimestamp</name>
                     </Parameter>
                     <Parameter>
                       <name>parameterLimits</name>
                     </Parameter>
               </ParameterList>
            </QueryData>
         </msg:Payload>
      </msg:ResponseMessage>
   </soap:Body>
</soap:Envelope>
```

B.4.2 Server Timestamp Request example

B.4.2.1 Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
      <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
            <msg:Verb>get</msg:Verb>
            <msg:Noun>QueryData</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
            <msg:Timestamp>2012-11-30T09:30:47.581Z</msg:Timestamp>
         </msg:Header>
         <msg:Request>
            <msg:Option>
               <msg:name>DataType</msg:name>
               <msg:value>serverTimestamp</msg:value>
            </msg:Option>
         </msg:Request>
      </msg:RequestMessage>
   </soap:Body>
</soap:Envelope>
```

B.4.2.2 Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
```

```
   <soap:Header/>
   <soap:Body>
      <msg:ResponseMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
            <msg:Verb>reply</msg:Verb>
            <msg:Noun>QueryData</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
            <msg:Timestamp>2012-11-30T09:30:48.004Z</msg:Timestamp>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      ...
      </Signature>
         </msg:Header>
         <msg:Reply>
            <msg:Result>OK</msg:Result>
         </msg:Reply>
           <msg:Payload>
            <QueryData xmlns="urn:iec62325.504:messages:1:0">
               <RequestParameters>
                  <Parameter>
                     <name>DataType</name>
                     <value>serverTimestamp</value>
                  </Parameter>
               </RequestParameters>
            </QueryData>
           </msg:Payload>
      </msg:ResponseMessage>
   </soap:Body>
</soap:Envelope>
```

B.4.3 Server Parameter Limits Request example

B.4.3.1 Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
      <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
            <msg:Verb>get</msg:Verb>
            <msg:Noun>QueryData</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
            <msg:Timestamp>2012-11-30T09:30:47.581Z</msg:Timestamp>
         </msg:Header>
         <msg:Request>
            <msg:Option>
               <msg:name>DataType</msg:name>
               <msg:value>parameterLimits</msg:value>
            </msg:Option>
         </msg:Request>
      </msg:RequestMessage>
   </soap:Body>
</soap:Envelope>
```

B.4.3.2 Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
      <msg:ResponseMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
```

```
            <msg:Verb>reply</msg:Verb>
            <msg:Noun>QueryData</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
            <msg:Timestamp>2012-11-30T09:30:48.004Z</msg:Timestamp>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      ...
      </Signature>
         </msg:Header>
         <msg:Reply>
            <msg:Result>OK</msg:Result>
         </msg:Reply>
         <msg:Payload>
            <QueryData xmlns="urn:iec62325.504:messages:1:0">
               <RequestParameters>
                  <Parameter>
                     <name>DataType</name>
                     <value>parameterLimits</value>
                  </Parameter>
               </RequestParameters>
               <ParameterList xmlns="urn:iec62325.504:messages:1:0">
                     <Parameter>
                       <name>MaxNumMessagesInListResponse</name>
                       <value>3000</value>
                     </Parameter>
                     <Parameter>
                       <name>MaxTimeIntervalInDaysInListRequest</name>
                       <value>3</value>
                     </Parameter>
               </ParameterList>
            </QueryData>
         </msg:Payload>
      </msg:ResponseMessage>
   </soap:Body>
</soap:Envelope>
```

B.4.4 Generic Query example

B.4.4.1 Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
      <msg:RequestMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
            <msg:Verb>get</msg:Verb>
            <msg:Noun>QueryData</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
            <msg:Timestamp>2012-11-30T09:30:47.581Z</msg:Timestamp>
         </msg:Header>
         <msg:Request>
            <msg:StartTime>2012-11-26T23:00:00Z</msg:StartTime>
            <msg:EndTime>2012-11-27T23:00:00Z</msg:EndTime>
            <msg:Option>
               <msg:name>DataType</msg:name>
               <msg:value>exampleWithParameters</msg:value>
            </msg:Option>
            <msg:Option>
               <msg:name>Parameter_1</msg:name>
               <msg:value>a value for parameter</msg:value>
            </msg:Option>
```

```
            <msg:Option>
               <msg:name>OtherParameter</msg:name>
               <msg:value>a value for other parameter</msg:value>
            </msg:Option>
         </msg:Request>
      </msg:RequestMessage>
   </soap:Body>
</soap:Envelope>
```

B.4.4.2 Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body>
      <msg:ResponseMessage xmlns:msg="http://iec.ch/TC57/2011/schema/message">
         <msg:Header>
            <msg:Verb>reply</msg:Verb>
            <msg:Noun>QueryData</msg:Noun>
            <msg:Context>PRODUCTION</msg:Context>
            <msg:Timestamp>2012-11-30T09:30:48.004Z</msg:Timestamp>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      ...
      </Signature>
         </msg:Header>
         <msg:Reply>
            <msg:Result>OK</msg:Result>
         </msg:Reply>
         <msg:Payload>
            <QueryData xmlns="urn:iec62325.504:messages:1:0">
               <RequestParameters>
                  <Parameter>
                     <name>DataType</name>
                     <value>exampleWithParameters</value>
                  </Parameter>
                  <Parameter>
                     <name>Parameter_1</name>
                     <value>a value for parameter</value>
                  </Parameter>
                  <Parameter>
                     <name>OtherParameter</name>
                     <value>a value for other parameter</value>
                  </Parameter>
               </RequestParameters>
               <n1:AnyKindOfDocument xmlns:n1="namespace declaration">
                  …
               </n1:AnyKindOfDocument>
             </QueryData>
         </msg:Payload>
      </msg:ResponseMessage>
   </soap:Body>
</soap:Envelope>
```

B.5 Fault

B.5.1 SOAP 1.2

The following shows a fault message returned by a “List” operation where the given TimeInterval is not valid, using SOAP 1.2:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
  <soap:Body>
    <soap:Fault>
      <soap:Code>
        <soap:Value>soap:Receiver</soap:Value>
      </soap:Code>
      <soap:Reason>
        <soap:Text xml:lang="es">EPF-04</soap:Text>
      </soap:Reason>
      <soap:Detail>
        <msg:FaultMessage
xmlns:msg="http://iec.ch/TC57/2011/schema/message">
          <msg:Reply>
            <msg:Result>FAILED</msg:Result>
            <msg:Error>
                <msg:code>EPF-04</msg:code>
                <msg:details>Invalid Time Interval: 2011-03aT22:00Z2011-03-
02T12:00Z </msg:details>
            </msg:Error>
          </msg:Reply>
        </msg:FaultMessage>
      </soap:Detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

B.5.2 SOAP 1.1

The following shows a fault message returned by a “List” operation where the given TimeInterval is not valid, using SOAP 1.1:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:Receiver</faultcode>
      <faultstring>EPF-04</faultstring>
      <detail>
        <msg:FaultMessage
xmlns:msg="http://iec.ch/TC57/2011/schema/message">
          <msg:Reply>
            <msg:Result>FAILED</msg:Result>
            <msg:Error>
                <msg:code>EPF-04</msg:code>
                <msg:details>Invalid Time Interval: 2011-03aT22:00Z2011-03-
02T12:00Z </msg:details>
            </msg:Error>
          </msg:Reply>
        </msg:FaultMessage>
      </detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

B.6 Digital signature

B.6.1 Basic example

This is an example of a response to the Put service:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope">
   <SOAP-ENV:Header>
   </SOAP-ENV:Header>
   <SOAP-ENV:Body>
       <msg:ResponseMessage
xmlns:msg="http://iec.ch/TC57/2011/schema/message">
            <msg:Header>
                <msg:Verb>reply</msg:Verb>
                <msg:Noun>Acknowledgement_MarketDocument</msg:Noun>
                <msg:Context>PRODUCTION</msg:Context>
                <msg:Timestamp>2014-04-15T13:06:29.885Z</msg:Timestamp>
         <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
            <SignedInfo>
               <CanonicalizationMethod
Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
               <SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"></SignatureMethod>
               <Reference URI="">
                  <Transforms>
                     <Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                     <Transform Algorithm="http://www.w3.org/TR/2001/REC-xml-
c14n-20010315"></Transform>
                  </Transforms>
                  <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                  <DigestValue>g87DHbL2vOP4N4sCUBjby/M6K48=</DigestValue>
               </Reference>
            </SignedInfo>

<SignatureValue>sl3aQ+UsBzesV6EQfGidB7Iuyk/iLIQP1VlKtyOEUEphGfP3y7no2tV9tw+mH2
FmId90hKCSurwnkakpKSryUNUDI9i5NPz+g+sI4N/hz0rVsSfh/CZV1ZaboVdVbngv/yc7JE77NFwR
e6kOAmWnUvnSywx162ZOasuEXxc+olkD+7HtPzAR86JhB2oAqqdtcuc+FGuMLnsCi6p5whZRq99rkh
vxnup3w2Y6b84nlJm/FJD9RsZJ0UInXVLMgALUmbxClKlKXwFUYLjSqY/SrFjFzkm0KshgWiFKOYhw
GhUy0+UNCpfRcGncOZj61tIuYD8bT205JITgSkdHO4bRmw==</SignatureValue>
            <KeyInfo>
               <X509Data>
                  <X509IssuerSerial>
                     <X509IssuerName>CN=testCA, O=test Domain</X509IssuerName>
                     <X509SerialNumber>131</X509SerialNumber>
                  </X509IssuerSerial>
                  <X509SubjectName> UID=myname, OU=intern, OU=myou, OU=users,
O=test, C=test</X509SubjectName>
                  <X509Certificate>
MIICOTCCAaKgAwIBAAIEU04yjTANBgkqhkiG9w0BAQUFADBhMQ0wCwYDVQQIEwRL
RVkhMQ8wDQYDVQQHEwZTRUNSRVQxDDAKBgNVBAoTA1RIRTEXMBUGA1UECxMOWU9V
IEhBVkUgRk9VTkQxGDAWBgNVBAMTD0NPTkdSQVRVTEFUSU9OUzAeFw0xNDA0MTYw
NzM0MzdaFw0xNTA4MjkwNzM0MzdaMGExDTALBgNVBAgTBEtFWSExDzANBgNVBAcT
BlNFQ1JFVDEMMAoGA1UEChMDVEhFMRcwFQYDVQQLEw5ZT1UgSEFWRSBGT1VORDEY
MBYGA1UEAxMPQ09OR1JBVFVMQVRJT05TMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCB
iQKBgQCQSgndj2T9o19c1anXRkLeiVnFc8K1BhthMddZQ6kMZWcrFhKdBmrA0Fok
JLYttpkVLcYHQcSP3+MygBItFpkhhoTPhPT2IjN4899cDGw10zG4fHjJF3Vn3S4v
+TiUKilb0yfVHPl2KqSgLd2X49yaNoc+97Rp2adYPuWxnLNHhwIDAQABMA0GCSqG
SIb3DQEBBQUAA4GBAAn55WsZTpnRBkDhP9EwsMAqOf+4nGzYn+zqeKrczico8ylj
9ndLFl71aFMxQLVd8u8EaW8kGbo92Chh4z4vm9y4AFCFeHM5EwBA8w9tOw5l7oU9
+NmMg6owGY6m+vP31623C/lbv66dQOHUXLO5H/VVTOUgBOyoGXBtaC1sGvm8
</X509Certificate>
               </X509Data>
            </KeyInfo>
         </Signature>
            </msg:Header>
```

```
            <msg:Reply>
                <msg:Result>OK</msg:Result>
            </msg:Reply>
            <msg:Payload>
<Acknowledgement_MarketDocument xmlns="urn:iec62325.351:tc57wg16:451-
1:acknowledgementdocument:6:0">
   <mRID>ACK_Schedule_D_20140416</mRID>
   <createdDateTime>2014-04-15T13:07:30Z</createdDateTime>
   <sender_MarketParticipant.mRID codingScheme="A01">10XEXAMPLE-EIC-P
</sender_MarketParticipant.mRID>
   <sender_MarketParticipant.marketRole.type>A04
</sender_MarketParticipant.marketRole.type>
   <receiver_MarketParticipant.mRID codingScheme="A01">10XEXAMPLE-EIC-P
</receiver_MarketParticipant.mRID>
   <receiver_MarketParticipant.marketRole.type>A04
</receiver_MarketParticipant.marketRole.type>
   <received_MarketDocument.mRID>Schedule_D_20140416
</received_MarketDocument.mRID>
   <received_MarketDocument.revisionNumber>1
</received_MarketDocument.revisionNumber>
   <received_MarketDocument.type>A51</received_MarketDocument.type>
   <received_MarketDocument.createdDateTime>2014-04-15T13:06:29Z
</received_MarketDocument.createdDateTime>
   <Reason>
      <code>A01</code>
   </Reason>
</Acknowledgement_MarketDocument>
            </msg:Payload>
         </msg:ResponseMessage>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Annex C
(informative)

Java code examples

C.1 Sign

The following simplified code excerpt signs a message following the rules described in this Technical Specification:

```
/**
* Signs the given xml document using the given private key and certificate.
* @param msgAsDocument The document to be signed, the result of the process
*   will be returned in this parameter.
* @param privateKey The private key to be used for signature.
* @param cert The certificate to be used for signature.
* @throws Exception If unable to sign the document.
* @see #signString(StringBuilder, RSAPrivateKey, X509Certificate)
*/
public static void signDocument(final Document msgAsDocument, final
RSAPrivateKey privateKey, final X509Certificate cert) throws Exception {

    try {
        XMLSignatureFactory fac = XMLSignatureFactory.getInstance("DOM");

        List<Transform> trfLst = new ArrayList<>();
            trfLst.add(fac.newTransform(Transform.ENVELOPED,
(TransformParameterSpec) null));

trfLst.add(fac.newCanonicalizationMethod(CanonicalizationMethod.INCLUSIVE,
(C14NMethodParameterSpec) null));

        Reference ref = fac
                .newReference("", fac.newDigestMethod(DigestMethod.SHA1,
null), trfLst, null, null);

        SignedInfo si =
fac.newSignedInfo(fac.newCanonicalizationMethod(CanonicalizationMethod.INCLUSI
VE, (C14NMethodParameterSpec) null),
fac.newSignatureMethod(SignatureMethod.RSA_SHA1, null),
                Collections.singletonList(ref));

        Node headerNode = null;
        NodeList nl =
msgAsDocument.getElementsByTagNameNS("http://iec.ch/TC57/2011/schema/message",
"Header");
        if (nl.getLength() == 1) {
            headerNode = nl.item(0);
        } else {
            throw new Exception("Invalid document.");
       }

        DOMSignContext dsc = new DOMSignContext(privateKey, headerNode);

        KeyInfoFactory keyInfoFactory = fac.getKeyInfoFactory();
        List<Object> x509Content = new ArrayList<>();

x509Content.add(keyInfoFactory.newX509IssuerSerial(cert.getIssuerDN().getName(
), cert.getSerialNumber()));
        x509Content.add(cert.getSubjectX500Principal().getName());
        x509Content.add(cert);
        X509Data xd = keyInfoFactory.newX509Data(x509Content);
```

```
        KeyInfo keyInfo =
keyInfoFactory.newKeyInfo(Collections.singletonList(xd));

        XMLSignature signature = fac.newXMLSignature(si, keyInfo);
        signature.sign(dsc);

    } catch (Exception e) {  //Simplified exception handling for brevity
        throw e;
    }
}
```

C.2 Verify

The following simplified code excerpt verifies a message's signature following the rules described in this Technical Specification:

```
/**
 * Verifies the signature of the given signed document.
 * @param msgAsDocument The document to be validated.
 * @throws Exception If the document cannot be validated
 *    or if its signature is invalid.
 */
public static void verifyDocument(final Document msgAsDocument) throws
Exception {

    try {
        Node signatureNode = null;
        NodeList nl = msgAsDocument.getElementsByTagNameNS(XMLSignature.XMLNS,
"Signature");
        if (nl.getLength() == 1) {
            signatureNode = nl.item(0);
        } else {
            throw new Exception("Invalid document.");
        }

        XMLSignatureFactory fac = XMLSignatureFactory.getInstance("DOM");
        DOMValidateContext valContext = new DOMValidateContext(new
KeyValueKeySelector(), signatureNode);
        XMLSignature signature = fac.unmarshalXMLSignature(valContext);
        boolean coreValidity = signature.validate(valContext);

        if (coreValidity == false) {
            StringBuilder sb = new StringBuilder("Signature validation failed.
Signature status: ");
            sb.append(signature.getSignatureValue().validate(valContext));
            sb.append(". ");
            Iterator<?> i =
signature.getSignedInfo().getReferences().iterator();
            for (int j = 0; i.hasNext(); j++) {
                boolean refValid = ((Reference)
i.next()).validate(valContext);
                sb.append("ref[").append(j).append("] validity status:
").append(refValid);
            }
            throw new Exception(sb.toString());
        }
    } catch (Exception e) {  //Simplified exception handling for brevity
        throw e;
    }
}
```

Annex D
(informative)

Regarding near real-time communications

The web services presented in this Technical Specification allow for the implementation of near real-time (or even real time) communications with responses in the range of seconds. However specific operational limits cannot and should not be set in stone in this Technical Specification as these may vary greatly depending on the specific business rules that apply to the exchanged payload.

For example: the time constraints of a continuous intraday market are completely different from a yearly capacity auction.

Parties responsible for managing systems with hard time constraints will need to make sure that their systems are fast enough to provide the necessary throughput. The communication layer, being direct communication from client to server, and being a synchronous invocation, will ensure that the additional delay introduced by the communication layer will be negligible, thus the term near real-time.

Note that of the different services presented, the preferred one for hard real-time requirements would be “Put” since the client has an immediate response to the submission of a payload. List and Get services used to poll data should be used when more relaxed deadlines are allowed.

The “Get” service can only guarantee near real-time in the sense that when a client requests a message, it will receive it with as small delay as possible, but not in the sense of it being part of a more complex communication (or dialog) involving several messages exchanged between client and server.

Annex E
(informative)

Regarding clients and servers configurations

The web services presented in this Technical Specification allow for different network configurations, where actors can be a client, a server, or both at the same time.

In the same way as with other International Standards that define client and server requirements, the fact that this Technical Specification defines the requirements for a client and a server does not mean that an actor aiming to comply with this specification needs to implement both the client and server requirements.

With this clarification in mind, when designing the communication requirements for a process where IEC TS 62325-504 will be used, the designers will be able to decide if all actors need to be both clients and servers as defined in IEC TS 62325-504, or there will be a reduced number of servers and the rest of the actors will only be clients.

This flexibility allows process designers to apply IEC TS 62325-504 tailoring it to their specific network configuration needs.

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
PO Box 131
CH-1211 Geneva 20
Switzerland

Tel: + 41 22 919 02 11
Fax: + 41 22 919 03 00
info@iec.ch
www.iec.ch