

DIN EN IEC 61968-100

ICS 29.240.01; 35.240.99

Ersatz für
DIN EN 61968-100:2014-08
Siehe Anwendungsbeginn

**Integration von Anwendungen in Anlagen der Elektrizitätsversorgung –
Systemschnittstellen für Netzführung –
Teil 100: Implementierungsprofile für die Anwendungsintegration
(IEC 61968-100:2022);
Englische Fassung EN IEC 61968-100:2022**

Application integration at electric utilities –
System interfaces for distribution management –
Part 100: IEC Implementation profiles for application integration
(IEC 61968-100:2022);
English version EN IEC 61968-100:2022

Intégration d'applications pour les services électriques –
Interfaces système pour la gestion de la distribution –
Partie 100: Profils de mise en oeuvre IEC pour l'intégration d'application
(IEC 61968-100:2022);
Version anglaise EN IEC 61968-100:2022

Gesamtumfang 257 Seiten

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE

Anwendungsbeginn

Anwendungsbeginn für die von CENELEC am 2022-03-23 angenommene Europäische Norm als DIN-Norm ist 2024-09-01.

Für DIN EN 61968-100:2014-08 besteht eine Übergangsfrist bis 2025-03-23.

Nationales Vorwort

Vorausgegangener Norm-Entwurf: E DIN EN IEC 61968-100:2022-08.

Für dieses Dokument ist das nationale Arbeitsgremium DKE/K 952 „Netzleittechnik“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE (www.dke.de) zuständig.

Aktuelle Informationen zu diesem Dokument können über die Internetseiten von DKE (www.dke.de) und DIN (www.din.de) durch eine Suche nach der Dokumentennummer aufgerufen werden.

Die enthaltene IEC-Publikation wurde vom IEC/TC 57 „Power systems management and associated information exchange“ erarbeitet und unterliegt dem Copyright der IEC .

Das IEC-Komitee hat entschieden, dass der Inhalt dieses Dokuments bis zu dem Datum (stability date) unverändert bleiben soll, das auf der IEC-Website unter „<https://webstore.iec.ch>“ zu diesem Dokument angegeben ist. Zu diesem Zeitpunkt wird entsprechend der Entscheidung des Komitees das Dokument

- bestätigt,
- zurückgezogen,
- durch eine Folgeausgabe ersetzt oder
- geändert.

Es werden die folgenden Schriftarten verwendet:

- für Programmfragmente oder XML-Elemente: `Courier New`.

Für den Fall einer undatierten Verweisung im normativen Text (Verweisung auf ein Dokument ohne Angabe des Ausgabedatums und ohne Hinweis auf eine Abschnittsnummer, eine Tabelle, ein Bild usw.) bezieht sich die Verweisung auf die jeweils aktuellste Ausgabe des in Bezug genommenen Dokuments.

Für den Fall einer datierten Verweisung im normativen Text bezieht sich die Verweisung immer auf die in Bezug genommene Ausgabe des Dokuments.

Der Zusammenhang der zitierten Dokumente mit den entsprechenden deutschen Dokumenten ergibt sich, soweit ein Zusammenhang besteht, grundsätzlich über die Nummer der entsprechenden IEC-Publikation.

Das Präsidium des DIN hat mit Präsidialbeschluss 1/2004 festgelegt, dass DIN-Normen, deren Inhalt sich auf internationale Arbeitsergebnisse der Informationsverarbeitung gründet, unter bestimmten Bedingungen allein in englischer Sprache veröffentlicht werden dürfen. Diese Bedingungen sind für den vorliegenden Norm-Entwurf erfüllt.

Da sich die Benutzer der vorliegenden Norm der englischen Sprache als Fachsprache bedienen, wird die Englische Fassung der EN IEC 61968-100 veröffentlicht. Zu deren Einleitung, Abschnitt 1, der den Anwendungsbereich festlegt, und Abschnitt 3, der die Begriffe und Abkürzungen festlegt, wurde eine Übersetzung angefertigt und als informativer Nationaler Anhang NA der vorliegenden Norm hinzugefügt. Für einige der verwendeten Abkürzungen existieren keine gebräuchlichen deutschen Benennungen, da sich die deutschen Anwender in der Regel ebenfalls der englischen Benennungen bedienen. Diese Norm steht nicht in unmittelbarem Zusammenhang mit Rechtsvorschriften und ist nicht als Sicherheitsnorm anzusehen.

Das Original-Dokument enthält Bilder in Farbe, die in der Papierversion in einer Graustufen-Darstellung wiedergegeben werden. Elektronische Versionen dieses Dokuments enthalten die Bilder in der originalen Farbdarstellung.

Änderungen

Gegenüber DIN EN 61968-100:2014-08 wurden folgende Änderungen vorgenommen:

- a) IEC 61968-100:2021 wurde überarbeitet, um Unklarheiten und Redundanzen zu beseitigen. Informative und normative Abschnitte sind als solche gekennzeichnet, und im letzteren Fall werden entsprechende verbale Formen der Sprache verwendet.
- b) Die durch IEC 61968-100:2021 definierten Nachrichten sind im Allgemeinen nicht rückwärtskompatibel mit denen aus der IEC 61968 100:2013. Annex I enthält die wesentlichen technischen Änderungen, die seit der Veröffentlichung von IEC 61968 100:2013 eingeführt wurden, und stellt geeignete Vorschläge für die Migration zur IEC 61968 100:2021 bereit.

Frühere Ausgaben

DIN EN 61968-100: 2014-08

Nationaler Anhang NA (informativ)

Auszugsweise Übersetzung

Reihenfolge und Inhalt der folgenden Abschnitte sind identisch mit denen in den jeweiligen Abschnitten der Englischen Fassung.

Einleitung

Dieses Dokument legt fest, wie Implementierungen von IEC 61968 unter Verwendung gängiger Technologien zur Unternehmensintegration interoperabel gemacht werden können. Insbesondere beschreibt es, wie die in IEC 61968 (Teil 3 bis Teil 9 und Teil 13), IEC 61970 und IEC 62325 definierten Nachrichten-Nutzlasten zwischen Systemen auszutauschen sind, die Transportmechanismen wie Webdienste und den Java-Nachrichten-Dienst nutzen.

Dieses Dokument kann auch über den für diese internationalen Normen festgelegten Informationsaustausch hinaus Anwendung finden, wie z. B. für die Integration von Marktsystemen oder für die allgemeine Unternehmensintegration.

1 Anwendungsbereich

1.1 Allgemeines

Dieser Teil von IEC 61968 definiert, wie Nachrichten zwischen kooperierenden Systemen ausgetauscht werden können, um die Übertragung von anwendungsspezifischen Daten zu ermöglichen. Solche anwendungsspezifischen Daten beinhalten unter anderem die in IEC 61968 (Teil 3 bis Teil 9 und Teil 13), IEC 61970 und IEC 62325 definierten Nutzdaten.

1.2 Über dieses Dokument

Dieses Dokument enthält normative Festlegungen für:

- einen Satz von Nachrichten-Archetypen (Abschnitt 5);
- einen Satz von Austauschmustern für Nachrichten, die sowohl vom sendenden als auch vom empfangenden System umzusetzen sind (Abschnitt 6);
- das genaue Format der Nachrichten, die über die verschiedenen Integrationstechnologien zu übermitteln sind, einschließlich einer genauen Beschreibung der Informationen, die jede Nachricht enthalten muss (Abschnitt 7);
- eine Reihe von Einschränkungen und Konventionen, die Anwendungen einhalten müssen, um den Nachrichtenaustausch mit IEC 61968-100 zu ermöglichen (Abschnitt 8);
- die Details dazu, wie IEC 61968-100-Nachrichten mit verschiedenen zugrunde liegenden Transportmechanismen implementiert werden sollten (Abschnitt 9).

1.3 Was nicht von diesem Dokument abgedeckt wird

Sicherheitsüberlegungen liegen außerhalb des Anwendungsbereichs von IEC 61968-100. Dieses Dokument verweist auf die Normenreihe IEC 62351 für Festlegungen und Praktiken in Bezug auf die sichere Übertragung von Nachrichten.

1.4 Zukünftige Überlegungen

1.4.1 Auswahl der Codierungsmechanismen

Dieses Dokument schreibt XML als normativen Kodierungsmechanismus für alle in diesem Dokument definierten Nachrichten vor.

Zukünftige Ausgaben von IEC 61968-100 können weitere normative Kodierungsmethoden festlegen, einschließlich der Unterstützung von JSON-Dokumenten (RFC 4627), deren Semantik durch die IEC CIM definiert ist und deren Syntax durch ein IETF-JSON-Schema definiert ist.

1.4.2 Wahl der Webdiensttechnologien

Dieses Dokument enthält normative Festlegungen für die Verwendung von SOAP-Webdiensten (9.2) und Java-Nachrichten-Diensten (9.3) für den Transport von Nachrichten.

Zukünftige Ausgaben von IEC 61968-100 können weitere normative Webdiensttechnologien wie REST festlegen.

3 Begriffe und Abkürzungen

3.1 Begriffe

Für die Anwendung dieser Spezifikation gelten die Begriffe nach IEC TS 61968-2:2011.

3.2 Abkürzungen

Die folgenden Abkürzungen werden in diesem Dokument verwendet:

AMI	erweiterte Infrastruktur für die Zählwerterfassung (en: Advanced Metering Infrastructure)
CIM	allgemeines Informationsmodell (en: Common Information Model)
CIS	Kundeninformationssystem (en: Customer Information System)
HES	Kopfstellensystem (en: Head End System)
HTTP	Hypertext-Übertragungsprotokoll (en: Hypertext Transfer Protocol)
IEC	Internationale elektrotechnische Kommission (en: International Electrotechnical Commission)
IETF	Internet Engineering Task Force
JMS	Java Message Service
JSON	JavaScript Object Notation
MDMS	Zählerdatenmanagementsystem (en: Meter Data Management System)
mRID	CIM Master Resource Identifier
RDF	Resource Description Framework
REST	REpresentational State Transfer
RFC	Bitte um Kommentare (en: Request For Comments)
SOAP	einfaches Objektzugriffsprotokoll (en: Simple Object Access Protocol)
URL	einheitlicher Ressourcenzeiger (en: Uniform Resource Locator)
WS	Webdienste (en: Web Services)
WSDL	Webdienst-Beschreibungssprache (en: Web Services Description Language)
XML	erweiterbare Auszeichnungssprache (en: eXtensible Markup Language)
XSD	XML-Schema Definition (en: XML Schema Definition)

– Leerseite –

EUROPEAN STANDARD

EN IEC 61968-100

NORME EUROPÉENNE

EUROPÄISCHE NORM

April 2022

ICS 33.200

Supersedes EN 61968-100:2013 and all of its amendments and corrigenda (if any)

English version

**Application integration at electric utilities – System
interfaces for distribution management – Part 100:
IEC Implementation profiles for application integration
(IEC 61968-100:2022)**

Intégration d'applications pour les services
électriques – Interfaces système pour la gestion
de la distribution – Partie 100: Profils de mise
en oeuvre IEC pour l'intégration d'application
(IEC 61968-100:2022)

Integration von Anwendungen in Anlagen der
Elektrizitätsversorgung – Systemschnittstellen für Netzführung –
Teil 100: Implementierungsprofile für die Anwendungsintegration
(IEC 61968-100:2022)

This European Standard was approved by CENELEC on 2022-03-23. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the CEN-CENELEC Management Centre or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the CEN-CENELEC Management Centre has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Bulgaria, Croatia, Cyprus, the Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Türkiye and the United Kingdom.



European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brüssel

European foreword

The text of document 57/2446/FDIS, future edition 2 of IEC 61968-100, prepared by IEC/TC 57 “Power systems management and associated information exchange” was submitted to the IEC-CENELEC parallel vote and approved by CENELEC as EN IEC 61968-100:2022.

The following dates are fixed:

- latest date by which the document has to be implemented at national level by (dop) 2022-12-23
publication of an identical national standard or by endorsement
- latest date by which the national standards conflicting with the document have (dow) 2025-03-23
to be withdrawn

This document supersedes EN 61968-100:2013 and all of its amendments and corrigenda (if any).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CENELEC shall not be held responsible for identifying any or all such patent rights.

This document has been prepared under a Standardization Request given to CENELEC by the European Commission and the European Free Trade Association.

Any feedback and questions on this document should be directed to the users' national committee. A complete listing of these bodies can be found on the CENELEC website.

Endorsement notice

The text of the International Standard IEC 61968-100:2022 was approved by CENELEC as a European Standard without any modification.

Annex ZA (normative)

Normative references to international publications with their corresponding European publications

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

NOTE 1 Where an International Publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

NOTE 2 Up-to-date information on the latest versions of the European Standards listed in this annex is available here: www.cenelec.eu.

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC 62325-301	2018	Framework for energy market communications – Part 301: Common information model (CIM) extensions for markets	EN IEC 62325-301	2018
IEC/TS 61968-2	2011	Application integration at electric utilities – System interfaces for distribution management – Part 2: Glossary	–	–
IEC 61968-9	2013	Application integration at electric utilities – System interfaces for distribution management – Part 9: Interfaces for meter reading and control	EN 61968-9	2014
IEC 61968-11	2013	Application integration at electric utilities – System interfaces for distribution management – Part 11: Common information model (CIM) extensions for distribution	EN 61968-11	2013
IEC 61970-301	2020	Energy management system application program interface (EMS-API) – Part 301: Common information model (CIM) base	EN IEC 61970-301	2020
IEC 61970-552	2016	Energy management system application program interface (EMS-API) – Part 552: CIMXML Model exchange format	EN 61970-552	2016
ISO/IEC 21320-1	2015	Information technology – Document Container File – Part 1: Core	–	–
ISO 8601	2004	Data elements and interchange formats - Information interchange - Representation of dates and times	–	–
Java Message Service (version 1.0.2 November 9, 1999)			–	–
RFC 1952	–	GZIP file format specification – Version 4.3	–	–
RFC 4122	–	A Universally Unique Identifier (UUID) URN Namespace	–	–

RFC 4648	-	The Base16, Base32, and Base64 Data Encodings	-	-
SOAP (Simple Object Access Protocol)			-	-

CONTENTS

European foreword	8
Annex ZA (normative) Normative references to international publications with their corresponding European publications	9
INTRODUCTION	25
1 Scope	26
1.1 General.....	26
1.2 About this document	26
1.3 What is not covered by this document	26
1.4 Future considerations	26
1.4.1 Choice of Encoding Mechanisms	26
1.4.2 Choice of Web Service Technologies	26
2 Normative references	27
3 Terms, definitions and abbreviated terms	27
3.1 Terms and definitions	27
3.2 Abbreviated terms	27
4 Overview	28
4.1 General.....	28
4.2 Message format.....	28
4.3 Profiles	29
4.4 Message archetypes	29
4.5 Header verb and noun	29
4.6 Example IEC 61968-100 messages.....	29
4.7 Message exchange patterns	30
4.7.1 General.....	30
4.7.2 Request/response message exchange pattern	30
4.7.3 Unsolicited event message exchange pattern	30
4.7.4 Consequential event message exchange pattern	31
4.8 Basic request/response message types	31
5 Message archetypes	32
5.1 General.....	32
5.2 Request message	32
5.2.1 General.....	32
5.2.2 Root element	32
5.2.3 Structure	32
5.2.4 Header element.....	33
5.2.5 Application-specific data	33
5.2.6 Side-effects of a request message.....	33

5.3 Response message 33

5.3.1 General 33

5.3.2 Root element 33

5.3.3 Structure 33

5.3.4 Header element 34

5.3.5 Application-specific data and status notifications 34

5.3.6 Simple acknowledgement message 34

5.4 Event message 35

5.4.1 General 35

5.4.2 Root element 35

5.4.3 Structure 35

5.4.4 Header element 35

5.4.5 Application-specific data 35

5.5 Fault message 35

5.5.1 General 35

5.5.2 Root element 36

5.5.3 Structure 36

5.5.4 Reply element 36

6 Elementary message exchange patterns 36

6.1 General 36

6.2 Request/response message exchange pattern 36

6.2.1 General 36

6.2.2 Response messages 37

6.3 Unsolicited event message exchange pattern 38

6.4 Consequential event message exchange pattern 38

6.5 Simple acknowledgement message exchange patterns 39

6.5.1 General 39

6.5.2 Request/response message exchange pattern with simple acknowledgement messages 40

6.5.3 Unsolicited event message exchange pattern with simple acknowledgement message 41

6.5.4 Consequential event message exchange pattern with simple acknowledgement messages 41

7 Message details 42

7.1 General 42

7.2 Message encoding 43

7.3 Schema definition and validation 43

7.4 Top-level structure 43

7.4.1 General 43

7.4.2 Structure of request messages 43

7.4.3 Structure of response messages 46

7.4.4 Structure of event messages 48

7.5	The <Header> element.....	51
7.5.1	General.....	51
7.5.2	Message header subelements	51
7.5.3	Message header for request messages.....	53
7.5.4	Message header for response messages	55
7.5.5	Message header for event messages	57
7.6	The <Reply> element.....	59
7.6.1	General.....	59
7.6.2	Structure.....	60
7.6.3	Reply subelements.....	60
7.6.4	The <Reply><Status> subelement.....	60
7.6.5	Combining the <Reply><Status><ObjectID>, <OperationSet>, <UpdateID> and <TransactionID> elements	69
7.6.6	The <Reply><Transaction> element.....	69
7.7	The <Payload> element.....	69
7.7.1	General.....	69
7.7.2	Payload subelements	69
7.7.3	The <Payload><OperationSet> element	70
7.7.4	The <Payload><ObjectUpdate> element	76
7.7.5	AbstractPayloadType data	78
7.7.6	Multiple abstract payloads.....	78
7.8	The <CompressedPayload> element.....	78
7.8.1	General.....	78
7.8.2	Structure.....	78
7.8.3	CompressedPayload elements	79
7.8.4	The <CompressedPayload><CompressionType> subelement	79
7.8.5	The <CompressedPayload><Base64> subelement.....	79
7.9	The Transaction Type	79
8	Constraints and conventions	80
8.1	General.....	80
8.2	Application-specific data	80
8.2.1	General.....	80
8.2.2	The AbstractPayloadType.....	81
8.2.3	Application-specific customisation	81
8.2.4	RDF data	81
8.3	Object management requests and notifications	81
8.3.1	General.....	81
8.3.2	Create request message	82
8.3.3	Change request message	82

8.3.4	Delete request message.....	83
8.3.5	Update request message	83
8.3.6	Object management notifications	84
8.4	Get requests	84
8.4.1	Making a get request	84
8.4.2	Specifying multiple constraints in a request message	85
8.4.3	Responding to a get request	85
8.5	Cancelling transactions	86
8.5.1	General	86
8.5.2	Message exchange pattern	86
8.5.3	Transactions IDs.....	87
8.5.4	Receiving a transaction ID.....	88
8.5.5	Sending a cancellation request message.....	88
8.5.6	Responding to a cancellation request message	88
8.6	Object identification	88
8.6.1	General	88
8.6.2	Rules for object identification.....	88
8.6.3	Identifying objects using an <mRID> element	89
8.6.4	Identifying objects using a <Name> element	89
8.6.5	Referencing objects	90
8.7	The ObjectIdentificationRevisionSets profile	91
8.7.1	General	91
8.7.2	Structure	91
8.7.3	Using the ObjectIdentificationRevisionSets profile	93
8.7.4	Adding one or more new identifiers	94
8.7.5	Removing one or more identifiers	94
8.7.6	Possible Errors Pertaining To ObjectIdentificationRevisionSets Requests	94
9	Transporting messages	94
9.1	General	94
9.2	Transporting over SOAP web services	95
9.2.1	General	95
9.2.2	Encoding of IEC 61968-100 messages inside SOAP messages.....	95
9.2.3	Example SOAP message	95
9.2.4	Mapping of IEC 61968-100 messages to SOAP messages	96
9.2.5	WSDL files	96
9.3	Transporting over JMS	97
9.3.1	General	97
9.3.2	Message exchange patterns	98
9.3.3	Application-level acknowledgement messages.....	98

9.3.4	JMS headers	98
	Annex A (normative) XSD files	100
A.1	General.....	100
A.2	The Message.xsd file	100
A.3	The AbstractPayloadType.xsd file.....	112
A.4	The LegacyObjectIdentificationRevisionSets.xsd File	113
A.5	The RDF.xsd file	119
	Annex B (normative) Template files	120
B.1	General.....	120
B.2	The GenericMessageWsdTemplate.txt file	120
B.3	The StrongMessageXsdTemplate.txt file.....	126
B.4	The StrongMessageWsdTemplate.txt file.....	139
	Annex C (informative) Example XSD profiles inheriting from AbstractPayloadType	146
C.1	GetMeterReadings profile	146
C.2	BinaryData profile.....	147
	Annex D (informative) Example IEC 61968-100:2021 messages.....	149
D.1	General.....	149
D.2	Messages dealing with object instantiation	149
D.2.1	General.....	149
D.2.2	Sequence diagram	149
D.2.3	Meter instantiation request (request message).....	150
D.2.4	Meter instantiation response (step 4, success case).....	152
D.2.5	Meter instantiation response (step 4, failure case)	152
D.3	Messages dealing with the event notification of object instantiation.....	153
D.3.1	General.....	153
D.3.2	Sequence diagram	153
D.3.3	Publication of a meter instantiation (event message)	153
D.3.4	Publication of a meter instantiation (simple acknowledgement message)	157
D.4	Messages dealing with requests for information concerning an object instantiation	157
D.4.1	General.....	157
D.4.2	Sequence diagram	157
D.4.3	Request for information concerning a meter instantiation (request message)	158
D.4.4	Request for information concerning a meter instantiation (response message)	159
D.5	Messages utilizing an operationset to enforce transactional integrity.....	160
D.5.1	General.....	160
D.5.2	Sequence diagram	160
D.5.3	Using an operationset to enforce transactional integrity (request message).....	161
D.5.4	Using an operationset to enforce transactional integrity (response message).....	164
D.6	Messages dealing with the publication of data	164

D.6.1	General	164
D.6.2	Sequence diagram	164
D.6.3	Publication of meter readings (event message)	165
D.6.4	Publication of meter readings (simple acknowledgement message)	166
D.7	Messages dealing with an on-demand data request resulting in a single response	167
D.7.1	General	167
D.7.2	Sequence Diagram	167
D.7.3	On-demand meter reading message (request message)	168
D.7.4	On-demand meter reading message (response message)	169
D.8	Messages dealing with an on-demand data request resulting in multiple responses	170
D.8.1	General	170
D.8.2	Sequence diagram	170
D.8.3	Request for an on-demand meter ping (request message)	171
D.8.4	Request for an on-demand meter ping (response message 1)	172
D.8.5	Request for an on-demand meter ping (response message 2)	173
D.8.6	Request for an on-demand meter ping (simple acknowledgement of message 2)	174
D.8.7	Request for an on-demand meter ping (response message 3)	175
D.8.8	Request for an on-demand meter ping (simple acknowledgement of message 3)	176
D.9	Messages dealing with historical data request	176
D.9.1	General	176
D.9.2	Sequence diagram	176
D.9.3	Historical meter read (request message)	177
D.9.4	Historical meter read (response message)	178
D.10	Messages dealing with historical data request (responses spread over several messages)	180
D.10.1	General	180
D.10.2	Historical meter read (response message 1 of 4)	180
D.10.3	Historical meter read (response message 2 of 4)	181
D.10.4	Historical meter read (response message 3 of 4)	181
D.10.5	Historical meter read (response message 4 of 4)	182
D.11	Messages dealing with multiple statuses within a response	183
D.11.1	General	183
D.11.2	Sequence diagram	184
D.11.3	Historical meter read (response message)	186
D.12	Messages dealing with a consequential event message	187
D.12.1	General	187
D.12.2	Sequence diagram	187
D.12.3	Request for a meter connect (request message)	188
D.12.4	Request for a meter connect (response message)	189
D.12.5	Request for a meter connect (event message)	190

D.12.6	Request for a meter connect (simple acknowledgement message)	191
D.13	Using two operationsets to sequence multiple operations with transactional integrity	191
D.13.1	General.....	191
D.13.2	Using two OperationSets to read two meters before disconnecting them (request message).....	192
D.13.3	Using two OperationSets to read two meters before disconnecting them (response message)	193
D.14	Messages dealing with object deletion.....	195
D.14.1	General.....	195
D.14.2	Sequence diagram	195
D.14.3	Request to delete a meter (request message).....	196
D.14.4	Request to delete a meter (response message).....	196
D.15	Messages dealing with cancellation of a single transaction	196
D.15.1	Meter control request message.....	196
D.15.2	Response message containing a single transaction id	197
D.15.3	Cancellation request (request message)	198
D.15.4	Cancellation request (response message)	198
D.16	Messages dealing with cancellation of multiple transactions.....	199
D.16.1	Response message containing multiple transaction IDs	199
D.16.2	Cancellation request (request message)	200
D.16.3	Cancellation request (response message)	201
D.17	Messages dealing with failed XSD validation	201
D.17.1	General.....	201
D.17.2	Message that fails XSD validation (request message)	202
D.17.3	Message that fails XSD validation (response message).....	202
D.18	Messages dealing with malformed requests.....	203
D.18.1	General.....	203
D.18.2	Message that is malformed (request message).....	203
D.18.3	Message that is malformed (response message)	204
Annex E (informative)	Example IEC 61968-100:2021 messages for updating objects	205
E.1	General.....	205
E.2	CreateMeterConfig.xml	205
E.3	ModifyFormNumber.xml	206
E.4	RemoveFormNumber.xml	207
E.5	AddMeterMultiplier.xml.....	207
E.6	ModifyMeterMultiplierKind.xml	208
E.7	ModifyMeterMultiplierValue.xml	209
E.8	RemoveMeterMultiplierValue.xml	210
E.9	RemoveMeterMultiplier.xml.....	211
E.10	AddMeterSeal.xml	212
E.11	AddMeterSealKind.xml.....	212

E.12	ModifyMeterSealCondition.xml.....	213
E.13	RemoveMeterSealCondition.xml.....	214
E.14	RemoveMeterSeal.xml	215
E.15	ModifyProgramId.xml.....	216
E.16	RemoveProgramId.xml.....	216
E.17	AddChannel.xml	217
E.18	ModifyChannelReadingType.xml	218
E.19	RemoveChannel.xml	219
E.20	AddConnectDisconnectFunction.xml	220
E.21	RemoveConnectDisconnectFunction.xml	221
E.22	DeleteMeterConfig.xml	222
E.23	DeleteMeterConfig.xml	222
E.24	CreateMeterConfigs.xml.....	223
E.25	MultipleObjectChanges.xml.....	225
E.26	MultipleObjectChangesWithOperationSet.xml	227
	Annex F (informative) Examples of ObjectIdentificationRevisionSets Messages	231
F.1	General	231
F.2	Add an mRID for the meter object.....	232
F.3	Modify name two of the meter object	233
F.4	Add name three of the meter object	234
F.5	Remove Name One of the Meter Object	236
F.6	Modify the mRID of the meter object.....	237
F.7	Remove the mRID of the meter object	239
F.8	Add an mRID and modify name two of the meter object.....	240
F.9	Modify name two of the meter object and add name one of the customer object.....	241
F.10	Add name two of the customer object to carry a new IdentifiedObject.name	244
	Annex G (informative) Example IEC 61968-100:2021 messages illustrating the cross-product rule	246
G.1	General	246
G.2	Specifying "OR" operations over the selection criteria	246
G.3	Specifying "AND" operations over the selection criteria	246
G.4	Specifying a combination of logical "AND" and "OR" operations	247
G.5	Example of multiple AbstractPayloadType profiles in a single message	248
	Annex H (informative) Validation of received messages	250
H.1	General	250
H.2	Example message validation using Altova XMLSpy	250
H.3	Example message validation using a Java validator.....	251
	Annex I (informative) Significant changes between IEC 61968-100:2013 and IEC 61968-100:2021.....	252
	Bibliography	257

Figure 1 – Structure of a request message	33
Figure 2 – Structure of a ResponseMessage	34
Figure 3 – Structure of an EventMessage	35
Figure 4 – Structure of a FaultMessage	36
Figure 5 – Request/response message exchange pattern	37
Figure 6 – Unsolicited event message exchange pattern	38
Figure 7 – Consequential event message exchange pattern	39
Figure 8 – Request/response message exchange pattern with simple acknowledgement messages	40
Figure 9 – Unsolicited message exchange pattern with simple acknowledgement messages	41
Figure 10 – Consequential event message exchange pattern with simple acknowledgement messages	42
Figure 11 – Structure of the <CancelRequestMessage> element	44
Figure 12 – Structure of the <ChangeRequestMessage> element	44
Figure 13 – Structure of the <CreateRequestMessage> element	44
Figure 14 – Structure of the <DeleteRequestMessage> element	45
Figure 15 – Structure of the <ExecuteRequestMessage> element	45
Figure 16 – Structure of the <GetRequestMessage> element	45
Figure 17 – Structure of the <UpdateRequestMessage> element	46
Figure 18 – Structure of the <CancelResponseMessage> element	46
Figure 19 – Structure of the <ChangeResponseMessage> element	47
Figure 20 – Structure of the <CreateResponseMessage> element	47
Figure 21 – Structure of the <DeleteResponseMessage> element	47
Figure 22 – Structure of the <ExecuteResponseMessage> element	48
Figure 23 – Structure of the <GetResponseMessage> element	48
Figure 24 – Structure of the <UpdateResponseMessage> element	48
Figure 25 – Structure of the <CanceledEventMessage> element	49
Figure 26 – Structure of the <ChangedEventMessage> element	49
Figure 27 – Structure of the <CreatedEventMessage> element	50
Figure 28 – Structure of the <DeletedEventMessage> element	50
Figure 29 – Structure of the <ExecutedEventMessage> element	50
Figure 30 – Structure of the <UpdatedEventMessage> element	50
Figure 31 – Structure of the <Header> element for request messages	54
Figure 32 – Structure of the <Header> element for response messages	56
Figure 33 – Structure of the <Header> element for event messages	58
Figure 34 – Structure of the <Reply> element	60
Figure 35 – Structure of the <Reply><Status> element	62
Figure 36 – Example of error indications in a <Reply> element	68
Figure 37 – Structure of the <Payload><OperationSet> element for request messages	71

Figure 38 – Structure of the <Payload><OperationSet> element for response messages	72
Figure 39 – Structure of the <Payload><OperationSet> element for event messages	72
Figure 40 – Structure of the <Payload><OperationSet><Operation> element for request messages	74
Figure 41 – Structure of the <Payload><OperationSet><Operation> element for response messages	75
Figure 42 – Structure of the <Payload><OperationSet><Operation> element for event messages	76
Figure 43 – Structure of the <Payload><ObjectUpdate> element	77
Figure 44 – Structure of the <CompressedPayload> element	79
Figure 45 – Structure of the TransactionType	80
Figure 46 – Message Exchange Pattern for a Cancellation Request	87
Figure 47 – General Naming Scheme	89
Figure 48 – Structure of an <ObjectID> element	90
Figure 49 – ObjectIdentificationRevisionSets profile	92
Figure 50 – Example SOAP message	95
Figure A.1 – Contents of Message.xsd file	112
Figure A.2 – Contents of AbstractPayloadType.xsd file	113
Figure A.3 – Contents of LegacyObjectIdentificationRevisionSets.xsd file	118
Figure A.4 – Contents of the RDF.xsd file	119
Figure B.1 – Contents of GenericMessageXsdTemplate.txt file	125
Figure B.2 – Contents of StrongMessageXsdTemplate.txt file	138
Figure B.3 – Contents of StrongMessageWsdTemplate.txt file	145
Figure C.1 – Example XSD profile illustrating inheritance from AbstractPayloadType	147
Figure C.2 – Example XSD profile suitable for the transmission of binary data	148
Figure D.1 – Example sequence diagram for a meter instantiation request	150
Figure D.2 – Example meter instantiation request message	152
Figure D.3 – Example meter instantiation response message (success case)	152
Figure D.4 – Example meter instantiation response message (failure case)	152
Figure D.5 – Example sequence diagram for the publication of a meter instantiation	153
Figure D.6 – Example publication of a meter instantiation (event message)	157
Figure D.7 – Example publication of a meter instantiation (simple acknowledgement message)	157
Figure D.8 – Example sequence diagram for requests for information concerning an object instantiation	158
Figure D.9 – Example request for information concerning a meter instantiation (request message)	159
Figure D.10 – Example request for information concerning a meter instantiation (response message)	160
Figure D.11 – Example sequence diagram for an OperationSet request	161
Figure D.12 – Example OperationSet message to enforce transactional integrity (request message)	163
Figure D.13 – Example OperationSet message to enforce transactional integrity (response message)	164

Figure D.14 – Example sequence diagram for the publication of meter readings	165
Figure D.15 – Example publication of routine meter readings (event message)	166
Figure D.16 – Example publication of routine meter readings (simple acknowledgement message)	167
Figure D.17 – Example sequence diagram for the retrieval of on-demand meter readings	168
Figure D.18 – Example on-demand meter readings request message	169
Figure D.19 – Example on-demand meter readings response message	170
Figure D.20 – Example sequence diagram for an on-demand meter ping request	171
Figure D.21 – Example on-demand meter ping request (request message)	172
Figure D.22 – Example on-demand meter ping request (response message)	173
Figure D.23 – Example on-demand meter ping request (response message 2)	174
Figure D.24 – Example on-demand meter ping request (simple acknowledgement message 2)	175
Figure D.25 – Example on-demand meter ping request (response message 3)	175
Figure D.26 – Example on-demand meter ping request (simple acknowledgement message 3)	176
Figure D.27 – Example sequence diagram for the retrieval of historical meter readings	177
Figure D.28 – Example historical meter read request (request message)	178
Figure D.29 – Example historical meter read request (response message)	179
Figure D.30 – Example historical meter read request (response message 1 of 4)	180
Figure D.31 – Example historical meter read request (response message 2 of 4)	181
Figure D.32 – Example historical meter read request (response message 3 of 4)	182
Figure D.33 – Example historical meter read request (response message 4 of 4)	183
Figure D.34 – Example sequence diagram for the retrieval of historical meter readings (request message)	184
Figure D.35 – Example historical meter read request, mixed success and failure cases (request message)	185
Figure D.36 – Example historical meter read request, mixed success and failure cases (response message)	187
Figure D.37 – Example sequence diagram for the requesting a meter connect	188
Figure D.38 – Example meter connect (request message)	189
Figure D.39 – Example meter connect (response message)	190
Figure D.40 – Example meter connect (event message)	191
Figure D.41 – Example meter connect (simple acknowledgement message)	191
Figure D.42 – Using two OperationSets to read two meters before disconnecting them (request message)	193
Figure D.43 – Using two OperationSets to Read two meters before disconnecting them (response message)	194
Figure D.44 – Example sequence diagram for the deletion of a meter	195
Figure D.45 – Example meter deletion (request message)	196
Figure D.46 – Example meter deletion (response message)	196
Figure D.47 – Example create(EndDeviceControls) request message	197
Figure D.48 – Example response message containing a single transaction ID	198
Figure D.49 – Example cancellation request message	198

Figure D.50 – Example response to a cancellation request message	199
Figure D.51 – Example response message containing several transaction IDs	200
Figure D.52 – Example cancellation request message	201
Figure D.53 – Example response to a cancellation request message	201
Figure D.54 – Example message that fails XSD validation (request message)	202
Figure D.55 – Example message that fails XSD validation (response message)	203
Figure D.56 – Example malformed message (request message)	203
Figure D.57 – Example malformed message (response message)	204
Figure E.1 – CreateMeterConfig.xml	206
Figure E.2 – ModifyFormNumber.xml	207
Figure E.3 – RemoveFormNumber.xml	207
Figure E.4 – AddMeterMultiplier.xml	208
Figure E.5 – ModifyMeterMultiplierKind.xml	209
Figure E.6 – ModifyMeterMultiplierValue.xml	210
Figure E.7 – RemoveMeterMultiplierValue.xml	211
Figure E.8 – RemoveMeterMultiplier.xml	211
Figure E.9 – AddMeterSeal.xml	212
Figure E.10 – AddMeterSealKind.xml	213
Figure E.11 – ModifyMeterSealCondition.xml	214
Figure E.12 – RemoveMeterDealCondition.xml	215
Figure E.13 – RemoveMeterSeal.xml	215
Figure E.14 – ModifyProgramId.xml	216
Figure E.15 – RemoveProgramId.xml	217
Figure E.16 – AddChannelId.xml	218
Figure E.17 – ModifyChannelReadingType.xml	219
Figure E.18 – RemoveChannel.xml	220
Figure E.19 – AddConnectDisconnectFunction.xml	221
Figure E.20 – RemoveConnectDisconnectFunction.xml	222
Figure E.21 – DeleteMeterConfig.xml	222
Figure E.22 – DeleteMeterConfig.xml	223
Figure E.23 – CreateMeterConfigs.xml	225
Figure E.24 – MultipleObjectChanges.xml	227
Figure E.25 – MultipleObjectChangesWithOperationSet.xml	230
Figure F.1 – Add an mRID for the Meter Object	233
Figure F.2 – Modify name two of the meter object	234
Figure F.3 – Add name three of the meter object	235
Figure F.4 – Remove name one of the meter object	237
Figure F.5 – Modify the mRID of the meter object	238
Figure F.6 – Remove the mRID of the meter object	239

Figure F.7 – Add an mRID and modify name two of the meter object	241
Figure F.8 – Add name one of the customer object and modify name two of the meter object.....	244
Figure F.9 – Add name two of the customer object to carry a new IdentifiedObject.name	245
Figure G.1 – Example get(MeterConfig) request illustrating a logical "OR" operation.....	246
Figure G.2 – Example get(MeterReadings) request illustrating a logical "AND" operation.....	247
Figure G.3 – Example request message illustrating combined logical "AND" and "OR" operations	248
Figure G.4 – Example of multiple AbstractPayloadType profiles in a single message	249
Figure H.1 – Example message suitably modified to specify the locations of the relevant XSD files	251
Figure H.2 – Example of local modifications made to the Message.xsd file	251
Table 1 – List of top-level root elements in request messages	43
Table 2 – List of top-level root elements in response messages	46
Table 3 – List of top-level root elements in event messages	49
Table 4 – Elements in the Message <Header> element.....	51
Table 5 – Verbs used in request messages	54
Table 6 – Verbs used in event messages	58
Table 7 – Subelements within the <Reply> element	60
Table 8 – Subelements within the <Reply><Status> element.....	63
Table 9 – Values of the <Reply><Status><Level> element.....	64
Table 10 – Normative values for the <Reply><Status><Code> element.....	64
Table 11 – Subelements within the <Payload> element.....	70
Table 12 – Subelements within the <Payload><OperationSet> element	71
Table 13 – Subelements within the <Payload><OperationSet><Operation> element	73
Table 14 – Subelements within the <Payload><ObjectUpdate> element	77
Table 15 – Subelements within the <Payload><ObjectUpdate><UpdateAction> element.....	78
Table 16 – Subelements within the <CompressedPayload> element	79
Table 17 – Values of the <CompressedPayload><CompressionType> element	79
Table 18 – Elements of a transaction type	80
Table 19 – Subelements within an <ObjectID> element.....	90
Table 20 – Elements of an ObjectIdentificationRevisionSets profile.....	93
Table 21 – Generic WSDL template variable names	96
Table 22 – Strongly-typed message XSD template variable names.....	97
Table 23 – Strongly-typed WSDL template variable names	97
Table F.1 – Initial naming states of the example meter object.....	231
Table F.2 – Initial naming states of the example customer object	231
Table F.3 – Add an mRID for the meter object.....	232
Table F.4 – Modify name two of the meter object	233
Table F.5 – Add name three of the meter object.....	235
Table F.6 – Remove name one of the meter object.....	236

Table F.7 – Modify the mRID of the meter object..... 238

Table F.8 – Remove the mRID of the meter object 239

Table F.9 – Add an mRID and modify name two of the meter object..... 240

Table F.10 – Modify name two of the meter object 242

Table F.11 – Add name one of the customer object..... 242

Table F.12 – Add name two of the customer object to carry a new IdentifiedObject.name 244

Table I.1 – Significant changes introduced in IEC 61968-100:2021 since IEC 61968-100:2013 253

Table I.2 – Detailed changes introduced from IEC 61968-100:2013 to IEC 61968-100:2021 254

Table I.3 – Elements removed from IEC 61968-100:2013 255

INTRODUCTION

This document specifies how implementations of IEC 61968 may be made interoperable using common enterprise integration technologies. In particular, it describes how message payloads defined in IEC 61968 (Parts 3 to 9 and Part 13), IEC 61970 and IEC 62325 are to be exchanged between systems using transport mechanisms such as web services and the Java Message Service.

This document may also find usage beyond information exchanges defined for these International Standards such as for the integration of market systems or for general enterprise integration.

APPLICATION INTEGRATION AT ELECTRIC UTILITIES – SYSTEM INTERFACES FOR DISTRIBUTION MANAGEMENT –

Part 100: IEC implementation profiles for application integration

1 Scope

1.1 General

This part of IEC 61968 defines how messages may be exchanged between cooperating systems in order to facilitate the transfer of application-specific data. Such application-specific data include but are not limited to the message payloads defined in IEC 61968 (Parts 3 to 9 and Part 13), IEC 61970 and IEC 62325.

1.2 About this document

This document provides normative definitions for:

- a set of message archetypes (Clause 5);
- a set of message exchange patterns that both sending and receiving systems are expected to implement (Clause 6);
- the exact format of the messages that are to be transmitted over the various integration technologies including a precise description of the information that each message must contain (Clause 7);
- a set of constraints and conventions to which applications must adhere in order to facilitate message exchange using IEC 61968-100 (Clause 8);
- the details of how IEC 61968-100 messages should be implemented using various underlying transport mechanisms (Clause 9).

1.3 What is not covered by this document

Security considerations lie outside the scope of IEC 61968-100. This document defers to the IEC 62351 series for definitions and practices relating to the secure transmission of messages.

1.4 Future considerations

1.4.1 Choice of Encoding Mechanisms

This document prescribes XML as the normative encoding mechanism for all messages defined by this document.

Future editions of IEC 61968-100 may specify additional normative encoding methods including support for JSON (RFC 4627) documents whose semantics are defined by the IEC CIM and whose syntax is defined by an IETF JSON schema.

1.4.2 Choice of Web Service Technologies

This document provides normative definitions for the use of SOAP Web Services (9.2) and Java Message Service (9.3) for the transport of messages.

Future editions of IEC 61968-100 may specify additional normative web service technologies such as REST.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 62325-301:2018, *Framework for energy market communications – Part 301: Common information model (CIM) extensions for markets*

IEC TS 61968-2:2011, *Application integration at electric utilities – System interfaces for distribution management – Part 2: Glossary*

IEC 61968-9:2013, *Application integration at electric utilities – System interfaces for distribution management – Part 9: Interfaces for meter reading and control*

IEC 61968-11:2013, *Application integration at electric utilities – System interfaces for distribution management – Part 11: Common information model (CIM) extensions for distribution*

IEC 61970-301:2020, *Energy management system application program interface (EMS-API) – Part 301: Common information model (CIM) base*

IEC 61970-552:2016, *Energy management system application program interface (EMS-API) – Part 552: CIMXML Model exchange format*

ISO IEC 21320-1:2015, *Information technology – Document Container File – Part 1: Core*

ISO 8601:2004, *Data elements and interchange formats – Information interchange – Representation of dates and times*

Java Message Service (version 1.0.2 November 9, 1999). Available from <https://docs.oracle.com/cd/E19957-01/816-5904-10/816-5904-10.pdf>

RFC 1952, GZIP file format specification version 4.3. Available from <https://tools.ietf.org/html/rfc1952>.

RFC 4122, A Universally Unique IDentifier (UUID) URN Namespace. Available from <https://tools.ietf.org/html/rfc4122>

RFC 4648, The Base16, Base32, and Base64 Data Encodings. Available from <https://tools.ietf.org/html/rfc4648>

SOAP (Simple Object Access Protocol). Available from <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this specification, the terms and definitions given in IEC TS 61968-2:2011 apply.

3.2 Abbreviated terms

The following abbreviated terms are used within this document:

AMI	Advanced Metering Infrastructure
CIM	Common Information Model
CIS	Customer Information System
HES	Head End System

HTTP	Hypertext Transfer Protocol
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
JMS	Java Message Service
JSON	JavaScript Object Notation
MDMS	Meter Data Management System
mRID	CIM master Resource IDentifier
RDF	Resource Description Framework
REST	REpresentational State Transfer
RFC	Request For Comments
SOAP	Simple Object Access Protocol
URL	Uniform Resource Locator
WS	Web Services
WSDL	Web Services Definition Language
XML	eXtensible Markup Language
XSD	XML Schema Definition

4 Overview

4.1 General

This clause provides an overview of this document. Details of the concepts described in this clause are provided in the subsequent clauses of this document.

4.2 Message format

An IEC 61968-100 message is an XML document. As such, it conforms to the `Message.xsd` XML Schema Definition (XSD) file that also forms part of this document (Clause A.2) and is capable of being successfully validated against it (Annex H).

Annex D, Annex E, Annex F and Annex G provide informative examples of IEC 61968-100 messages. These examples illustrate that an IEC 61968-100 message comprises both *header* information and *application-specific data*.

The header information is contained within a `<Header>` element inside the message. This header provides meta-data that describe how the remainder of the message is to be interpreted and processed. For example, the header shows whether the message is a command, a request for data, a reply to a previous message or an asynchronous notification.

The application-specific data are located within the `<Reply>` or `<Payload>` elements, depending on the purpose of the message. The `<Reply>` element conveys the status such as success, warning or error indications associated with a previous request back to the requesting system. The `<Payload>` element contains application-specific data conforming to profiles defined in other IEC standards.

This document makes no attempt to define these application-specific data. They are left to be defined by the relevant IEC working groups or the local organisation. However, this document imposes certain constraints and conventions to which the messages conveying such data shall adhere (Clause 8).

4.3 Profiles

The application-specific data generally conform to a profile. A profile is an XML schema definition (XSD) that has been defined by the corresponding IEC working group or local organisation. A profile is typically but not necessarily derived from the Common Information Model (CIM) as defined in standards such as IEC 62325-301, IEC 61968-11 and IEC 61970-301.

A profile generally falls into one of the following categories:

- It may correspond to a class, possibly with other associated classes, that belong to the CIM and which are related to one another by means of inheritance or association.

EXAMPLE 1 An IEC 61968-9:2013 MeterReadings profile holds information relating to the data read from a smart meter. MeterReading is a class defined in the CIM.

EXAMPLE 2 An IEC 61968-9:2013 CustomerConfig profile holds information pertaining to customer management. Customer is a CIM class but CustomerConfig is not.

- A profile may describe the linkages among CIM objects rather than the objects themselves.

EXAMPLE 3 An IEC 61968-9:2013 MasterDataLinkageConfig profile specifies the relationship between various CIM objects such as Customer, Meter, UsagePoint and so on. However, MasterDataLinkageConfig is not itself a CIM class.

- A profile may describe a set of query parameters by which a request for one or more existing CIM objects may be made. In this document these are referred to as application-specific query parameters.

EXAMPLE 4 An IEC 61968-9:2013 GetMeterReadings profile allows query parameters to be defined by which data read from one or more meters may be retrieved.

- A local organisation may define its own set of profiles. These have meaning only within that organisation.

4.4 Message archetypes

Messages are categorised according to whether they are request messages, response messages or event (asynchronous notification) messages. These categories are termed the message archetypes (Clause 5).

4.5 Header verb and noun

The message archetype can be inferred from the header verb which is carried in the `<Header><Verb>` element. The verb is one of a fixed set of enumerated verbs that are listed in clauses 7.5.3.3, 7.5.4.3 and 7.5.5.3.

A profile has a name by which it is identified in the message header – for example, MeterReadings, CustomerConfig or MasterDataLinkageConfig. This name is termed the header noun and is located in the `<Header><Noun>` element. (The precise rules for specifying the noun are more complicated than this brief description. Subclauses 7.5.3.4, 7.5.4.4 and 7.5.5.4 provide the full details).

The header verb and header noun together determine the purpose of the message. They are designated by the terminology `verb(noun)`.

4.6 Example IEC 61968-100 messages

Subclause D.7.3 illustrates a simple IEC 61968-100 message that shows the `<Header>` and `<Payload>` elements. In this example, the application-specific data in the `<Payload>` element are defined according to IEC 61968-9:2013 and form a request for retrieving meter read data.

Subclause D.7.4 depicts the corresponding response message and shows an IEC 61968-100 message with `<Header>`, `<Reply>` and `<Payload>` elements. The `<Reply>` element contains a status notification, in this

case a notification of success. The `<Payload>` element contains the meter read data that are being returned as the result of the previous request. These application-specific data are also defined according to IEC 61968-9:2013.

4.7 Message exchange patterns

4.7.1 General

This document defines several basic kinds of message exchange patterns (Clause 6). Any exchange of IEC 61968-100 messages between cooperating systems is composed of one or more of these message exchange patterns.

4.7.2 Request/response message exchange pattern

The request/response message exchange pattern (6.2) is used when a requesting system needs to request data from or that an action be taken by a responding system. The responding system processes the request and sends one or more response messages back to the requesting system.

The `<Header><Verb>` in the request message must be one of `cancel`, `create`, `delete`, `change`, `execute`, `get` or `update`. The header verb in the response message is always `reply`.

EXAMPLE 1 An IEC 61968-9:2013 `create(CustomerConfig)` message represents a request from one system to another to instantiate a CIM Customer object using the data supplied in the payload. The noun in this message is `CustomerConfig` and the `CustomerConfig` profile is used in the payload section. On receipt of such a message, the receiving system processes it and then sends a `reply(CustomerConfig)` message back to the request originator. This response message shows the status of the create request.

Messages that illustrate a similar example may be found in Clause D.2.

In distinction to the conventional request/response paradigm, this document permits multiple response messages to be associated with a single request. This may be used in cases where larger response payloads need to be separated into multiple messages or where there are multiple operations occurring in parallel and independently of one another.

EXAMPLE 2 An IEC 61968-9:2013 `get(MeterReadings)` request denotes that the given message is a request for specific meter reading data. The name of the profile in the request message is `GetMeterReadings`. The name of the profile in the corresponding reply is `MeterReadings`. The receipt of a `get(MeterReadings)` request can cause a head end system (HES) to request data from meters in its AMI network. There may be some delay as the data from the meters are returned back to the HES. The HES may decide to wait until data from all meters are received before bundling these together into a single `reply(MeterReadings)` message back to the request originator. Alternatively, it may elect to send data back to the requestor as each meter responds separately over the AMI network. In the latter case, the single request message may result in multiple `reply(MeterReadings)` messages being generated and sent to the requestor.

4.7.3 Unsolicited event message exchange pattern

The unsolicited event message exchange pattern (6.3), sometimes known as 'fire and forget', consists of a single message sent from one system to another and for which no response is expected. This may occur when the message is sent as the result of some autonomous action within a system. Such a message is termed an unsolicited event message.

The `<Header><Verb>` in an unsolicited event message belongs to one of the set `canceled`, `created`, `changed`, `deleted`, `executed` or `updated`.

EXAMPLE A HES may send out an IEC 61968-9:2013 `created(EndDeviceEvents)` message as a result of a meter suffering a power outage or in order to send a high temperature alarm.

A full example may be found in Clause D.6.

4.7.4 Consequential event message exchange pattern

The consequential event message exchange pattern (6.4) comprises the request/response followed by the unsolicited event message exchange pattern. In this case, the associated event message is termed a consequential event message. The complete sequence of messages comprises the original request, a single response that denotes the status of that request and then zero or more consequential event messages.

EXAMPLE A meter data management system (MDMS) may send an IEC 61968-9:2013 `create(EndDeviceControls)` message to a HES to request an operation be performed on meters connected to its AMI network. The HES verifies the request and returns a `reply(EndDeviceControls)` message back to the MDMS. This response message denotes whether the create command can be carried out successfully or whether it contains errors – for example, references to non-existent meters – or other informational conditions. It then sends the appropriate commands over the AMI network to the meters. When these react to the commands, they respond back to the HES with the results of the command. The HES in turn sends one or more `created(EndDeviceEvents)` messages back to the MDMS to inform it that the meters have changed state.

A full example may be found in Clause D.12.

4.8 Basic request/response message types

An IEC 61968-100 message appearing in a request/response message exchange pattern falls into one of a set of categories according to its `<Header><Verb>`.

- A message with a verb of `create` is a command to instantiate a new object on the receiving system. The act of instantiating an object may, as a side-effect, also cause the receiving system to perform some external action.

EXAMPLE 1 A customer information system (CIS) may send information to a MDMS using an IEC 61968-9:2013 `create(CustomerConfig)` message. The MDMS then uses the supplied information to instantiate the new customer in its local database.

EXAMPLE 2 An MDMS may send an IEC 61968-9:2013 `create(MeterReadSchedule)` message to a HES to request it to perform some future meter reading operations.

- A message where the verb is `change` is a command to replace an existing object in its entirety on the receiving system. This may cause some external actions to be performed.

EXAMPLE 3 An IEC 61968-9:2013 `change(CustomerConfig)` message may be used to instruct a CIS to replace the information in its local database.

- A message where the verb is `delete` is a command to completely remove an existing object from the receiving system. This may cause some external actions to be performed.

EXAMPLE 4 An IEC 61968-9:2013 `delete(CustomerConfig)` message may be used to instruct a CIS to remove the corresponding customer entries from its local database.

EXAMPLE 5 An MDMS may send an IEC 61968-9:2013 `delete(MeterReadSchedule)` message to a HES to request it to remove a meter reading schedule.

- A message with a verb of `get` is used to query a receiving system thereby causing one or more objects to be returned. The message supplies query parameters in its `<Payload>` section and the receiving system returns the requested data accordingly.

EXAMPLE 6 An IEC 61968-9:2013 `get(CustomerConfig)` message may cause a CIS to return information about the requested customers from its local database.

- A message with a verb of `execute` is used to group several commands together into one message. Transactional integrity may be optionally enforced such that either all the commands succeed or none of them do. The verb `execute` is used only in conjunction with the noun `OperationSet` (7.5.3, 7.7.3).

- A message with a verb of `cancel` is a command for cancelling an operation in progress.

EXAMPLE 7 An IEC 61968-9:2013 `cancel(EndDeviceControls)` message may be used to cancel a pending operation resulting from a previous `create(EndDeviceControls)` message.

- A message with a verb of `update` is a command to make partial modifications to an existing object (8.3.5). The `update` verb is new to this edition of IEC 61968-100.

EXAMPLE 8 An IEC 61968-9:2013 `update(CustomerConfig)` message may be used to instruct a CIS to update selected information such as a customer's telephone number in its local database.

- A message with a verb of `reply` is used for one system to inform another that it has received a command or request in a message containing one of the other verbs. The response message may include any requested data as well as any application status notifications.

EXAMPLE 9 An IEC 61968-9:2013 `reply(MeterReadings)` message may be sent from a HES to a MDMS to provide meter readings in response to a previous IEC 61968-9:2013 `get(MeterReadings)` request. It may also include error notifications – for instance, if the original request specifies invalid meters.

5 Message archetypes

5.1 General

This document defines four broad kinds of message: request messages, response messages, event messages and fault messages (5.2, 5.3, 5.4, 5.5). These message kinds are termed the message archetypes.

Every IEC 61968-100 message shall conform to one of these archetypes.

5.2 Request message

5.2.1 General

An IEC 61968-100 request message shall be used for sending a request from a requesting system to a responding system.

A request message shall be used solely within the request/response (6.2) or the consequential event (6.4) message exchange patterns.

5.2.2 Root element

The name of the root element of a request message depends on the value of the `<Header><Verb>` element in the message (7.5.3.3) and shall be taken from one of the root element names listed in Table 1.

This rule shall not apply in the case of strongly-typed web services (9.2.5).

5.2.3 Structure

The structure of a request message is depicted in Figure 1.

NOTE The generic name `RequestMessage` appearing in Figure 1 denotes one of the request element names listed in Table 1.

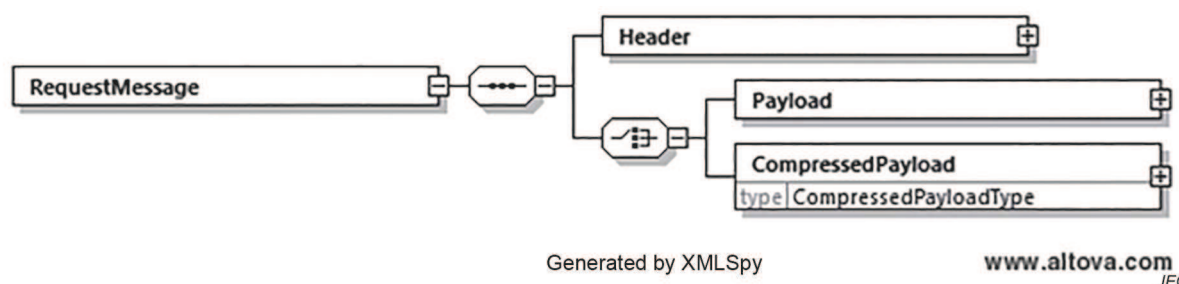


Figure 1 – Structure of a request message

A request message shall contain a `<Header>` subelement (7.5) together with either a `<Payload>` subelement (7.7) or a `<CompressedPayload>` subelement (7.8).

5.2.4 Header element

Subclause 7.5.2 specifies the `<Header>` element as it pertains to request messages.

5.2.5 Application-specific data

Any application-specific data that are associated with the request shall be carried in the `<Payload>` or `<CompressedPayload>` subelement (7.7, 7.8).

5.2.6 Side-effects of a request message

The reception of an request message by a responding system may cause that system to carry out some application-specific actions. Such actions or side-effects are non-normative and are beyond the scope of this document. They are left to be defined by the relevant working groups and application developers.

5.3 Response message

5.3.1 General

An IEC 61968-100 response message shall be used for sending a reply from a responding system to a requesting system that has previously sent a request message.

A response message shall be used within the request/response (6.2) or the consequential event (6.4) message exchange patterns. A response message may also be used for the acknowledgement of a message receipt where required by the underlying transport (6.5).

5.3.2 Root element

The name of the root element of a response message depends on the value of the `<Header><Verb>` element of the original request message (7.5.3.3) and shall be taken from one of the root element names listed in Table 2.

This rule shall not apply in the case of strongly-typed web services (9.2.5).

5.3.3 Structure

The structure of a response message is depicted in Figure 2.

NOTE The generic name `ResponseMessage` appearing in Figure 2 denotes one of the response element names listed in Table 2.

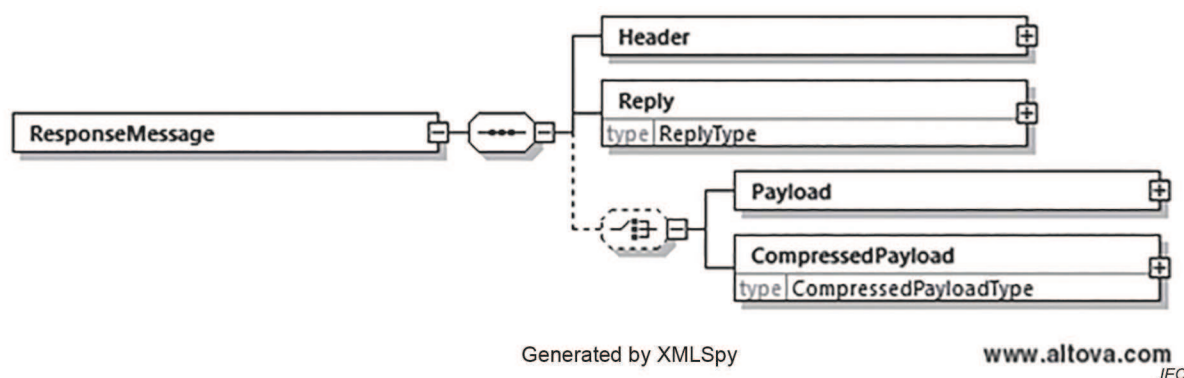


Figure 2 – Structure of a ResponseMessage

A response message shall contain both `<Header>` and `<Reply>` subelements (7.5, 7.6). It may also contain a `<Payload>` subelement (7.7) or a `<CompressedPayload>` subelement (7.8).

5.3.4 Header element

Subclause 7.5.2 specifies the `<Header>` element as it pertains to response messages.

5.3.5 Application-specific data and status notifications

A response message shall contain status indications. It may also contain application-specific data.

The application-specific data, if present, shall be placed in the `<Payload>` (7.7) or `<CompressedPayload>` subelement (7.8).

Any status indications shall be placed in the `<Reply><Status>` elements (7.6.4).

In the case of a response message returning notifications of success and no informational notifications, a `<Reply>` element containing an explicit indication of success shall be included in the message (7.6.4.3).

EXAMPLE 1 D.4.3 provides an example of a response message being used to convey application-specific data.

EXAMPLE 2 D.11.3 provides an example of a response message being used to convey both application-specific data and error notifications.

5.3.6 Simple acknowledgement message

A simple acknowledgement message is a specialised response message that is used solely to convey the receipt of a message (6.5).

The following elements shall be set in a simple acknowledgement message:

- The `<Header><Verb>` shall be `reply`.
- The `<Header><Noun>` shall have the same value as that of the original message for which the simple acknowledgement is being sent.
- A single `<Reply><Status>` element shall be included in the simple acknowledgement message (7.6.4). This shall contain `<Reply><Status><Level>` and `<Reply><Status><Code>` subelements with the values `OK` and `0.3` respectively.
- It may also contain an optional `<Reply><Status><Reason>` element containing some arbitrary text.

No application-specific data shall be conveyed inside a simple acknowledgement message.

5.4 Event message

5.4.1 General

An IEC 61968-100 event message shall be used for sending data from one system to another when no response is expected or allowed. It shall be used for the publication of data or for the notification of an occurrence or incident.

An event message shall be sent either in the context of the unsolicited event (6.3) or the consequential event (6.4) message exchange patterns.

5.4.2 Root element

The name of the root element of an event message depends on the value of the `<Header><Verb>` element in the message (7.5.5.3) and shall be taken from one of the enumerated values listed in Table 3.

This rule shall not apply in the case of strongly-typed web services (9.2.5).

5.4.3 Structure

The structure of an event message is depicted in Figure 3.

NOTE The generic name EventMessage appearing in Figure 3 denotes one of the element names listed in Table 3.

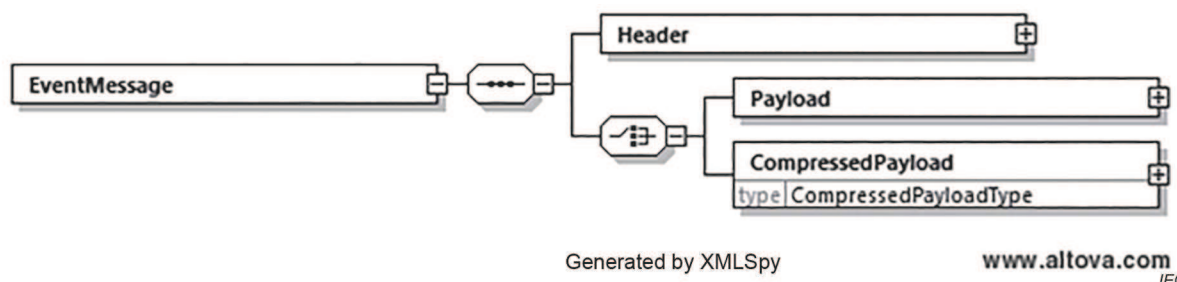


Figure 3 – Structure of an EventMessage

An event message shall contain a `<Header>` subelement (7.5) together with either a `<Payload>` subelement (7.7) or a `<CompressedPayload>` subelement (7.8).

5.4.4 Header element

Subclause 7.5.2 specifies the `<Header>` element as it pertains to event messages.

5.4.5 Application-specific data

The application-specific data shall be placed in the `<Payload>` element (7.7.3).

EXAMPLE D.3.3 provides an example of an event message.

5.5 Fault message

5.5.1 General

An IEC 61968-100 fault message shall be used solely in the case where a recipient system has received a malformed request message; that is, a request message that cannot be parsed as a valid XML document. In such a case, the responding system shall return a fault message to the requesting system.

A fault message shall not be used for any other kind of error detected by the recipient system.

XML schema validation errors shall invoke a response message containing a `<Reply><Status><Code>` element with an appropriate value (7.6.4.7).

EXAMPLE D.18.3 provides an example of a fault message.

5.5.2 Root element

A fault message shall contain `<FaultMessage>` as its root element. This is depicted in Figure 4.

5.5.3 Structure

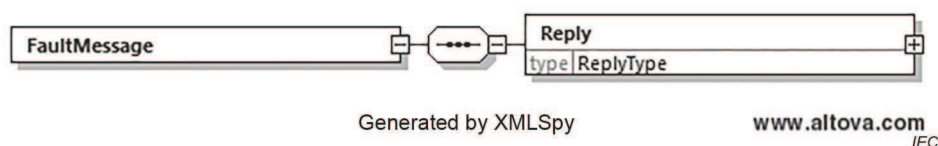


Figure 4 – Structure of a FaultMessage

5.5.4 Reply element

Subclause 7.6.4.5 specifies the `<Reply>` element as it pertains to fault messages

6 Elementary message exchange patterns

6.1 General

This clause describes the set of IEC 61968-100 elementary message exchange patterns. Any exchange of IEC 61968-100 messages between cooperating systems shall be composed of one or more of these elementary message exchange patterns.

6.2 Request/response message exchange pattern

6.2.1 General

A request/response message exchange pattern shall consist of a single request message (5.2) sent from a requesting system to a responding system followed by one or more response messages (5.3) sent from the responding system to the requesting system.

EXAMPLE D.1 provides an example of a message exchange based on the request/response message exchange pattern.

The request/response message exchange pattern is depicted in Figure 5.

6.3 Unsolicited event message exchange pattern

An unsolicited event message exchange pattern shall be used to send a notification from one system to another in those cases when no application-level response is expected or required. An unsolicited event message exchange pattern shall consist of a single event message (5.4).

An unsolicited event message shall be sent as a result of some autonomous action in a system and not as the direct result of a previous request.

The unsolicited event message exchange pattern is depicted in Figure 6.

EXAMPLE D.6 provides an example of an unsolicited event message.

If a sending system sends a message to a receiving system that is intended to cause some operation to take place and the sending system needs to know whether that operation succeeds or fails, then it shall use the request/response message exchange pattern and not the unsolicited event message exchange pattern.

An unsolicited event message should be sent only when the underlying operation has completed.

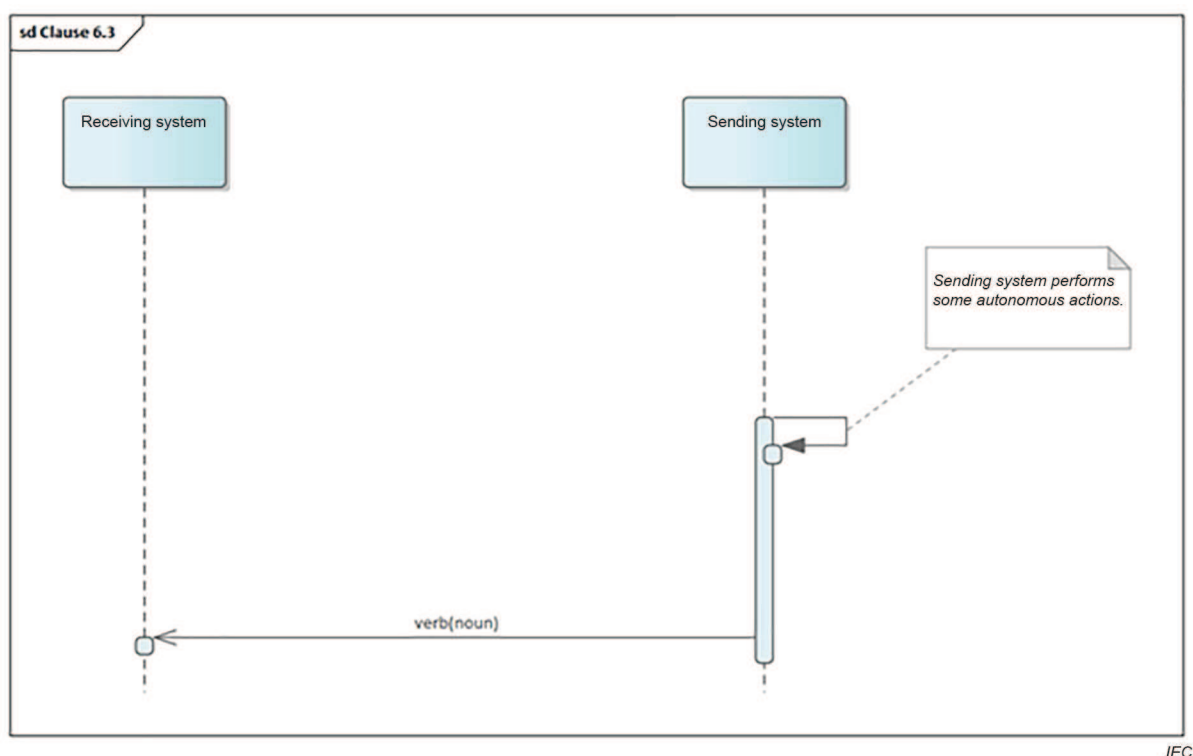


Figure 6 – Unsolicited event message exchange pattern

6.4 Consequential event message exchange pattern

The consequential event message exchange pattern shall consist of a request message (5.2) followed by a single response message (5.3) followed by zero or more event messages (5.4).

A consequential event message exchange pattern differs from the unsolicited event message exchange pattern in that the event being communicated is the result of a previous request message.

The consequential event message exchange pattern is depicted in Figure 7.

EXAMPLE D.12 provides an example of a message exchange based on the consequential event message pattern.

NOTE The messages being conveyed in a request/response message exchange pattern share the same <Header><Noun> element (7.5.2). For the consequential event message pattern, however, the <Header><Noun> element of the event messages is different to those of the initial request/response. Furthermore, the <Header><Verb> element of the event messages is not "reply".

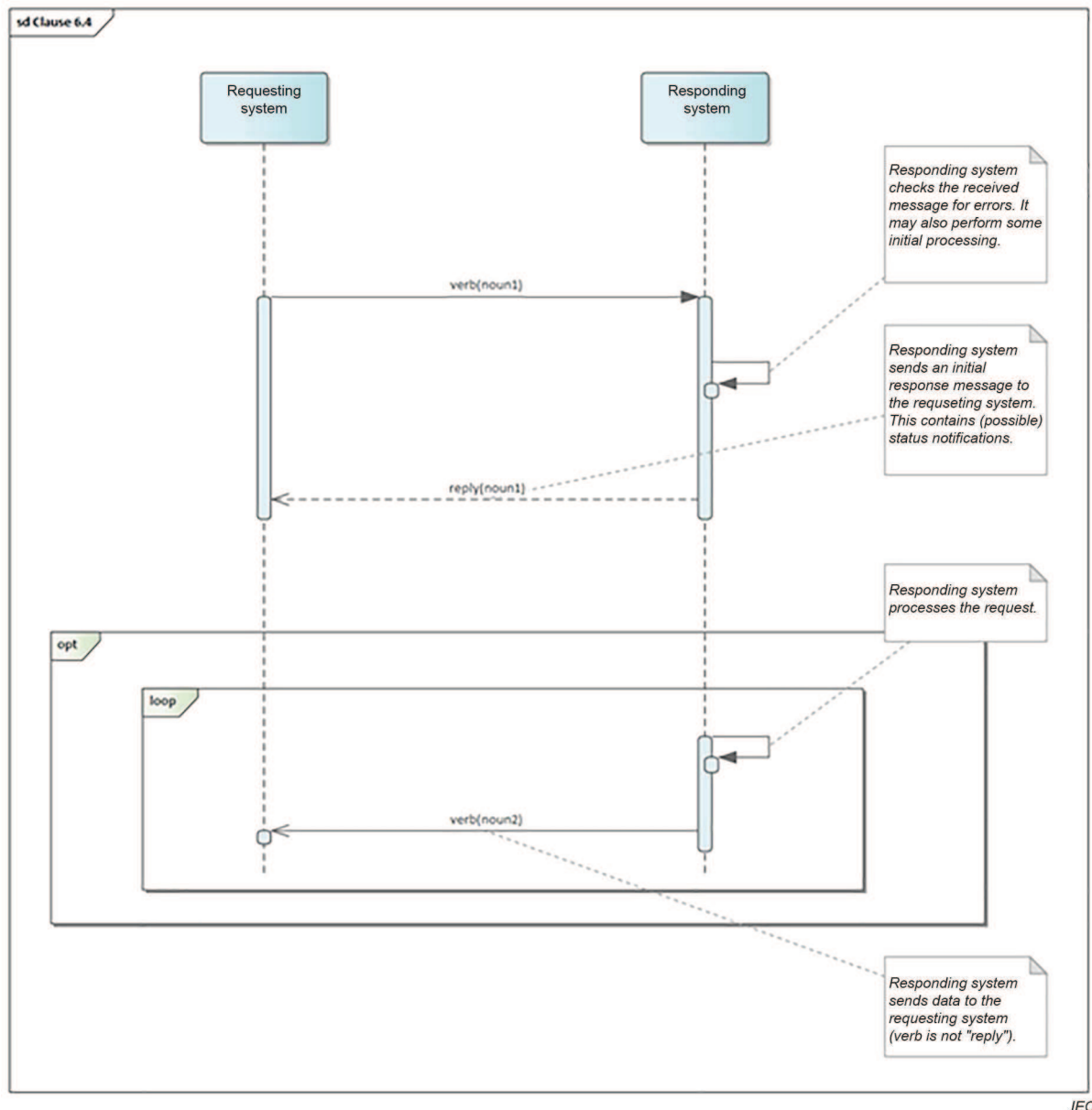


Figure 7 – Consequential event message exchange pattern

6.5 Simple acknowledgement message exchange patterns

6.5.1 General

A responding system shall send a simple acknowledgement message in either one of the following circumstances:

- The responding system has received a request message where the `<Header><AckRequired>` element is present and has the value `true` (7.5.2).
- The responding system has received a response message (5.2) or an event message (5.4) and the underlying transport requires that a transport-level acknowledgement of the response or event message be sent (9.2.4).

6.5.2 Request/response message exchange pattern with simple acknowledgement messages

The request/response message exchange pattern of Figure 5 is modified as shown in Figure 8 when a simple acknowledgement message is to be sent.

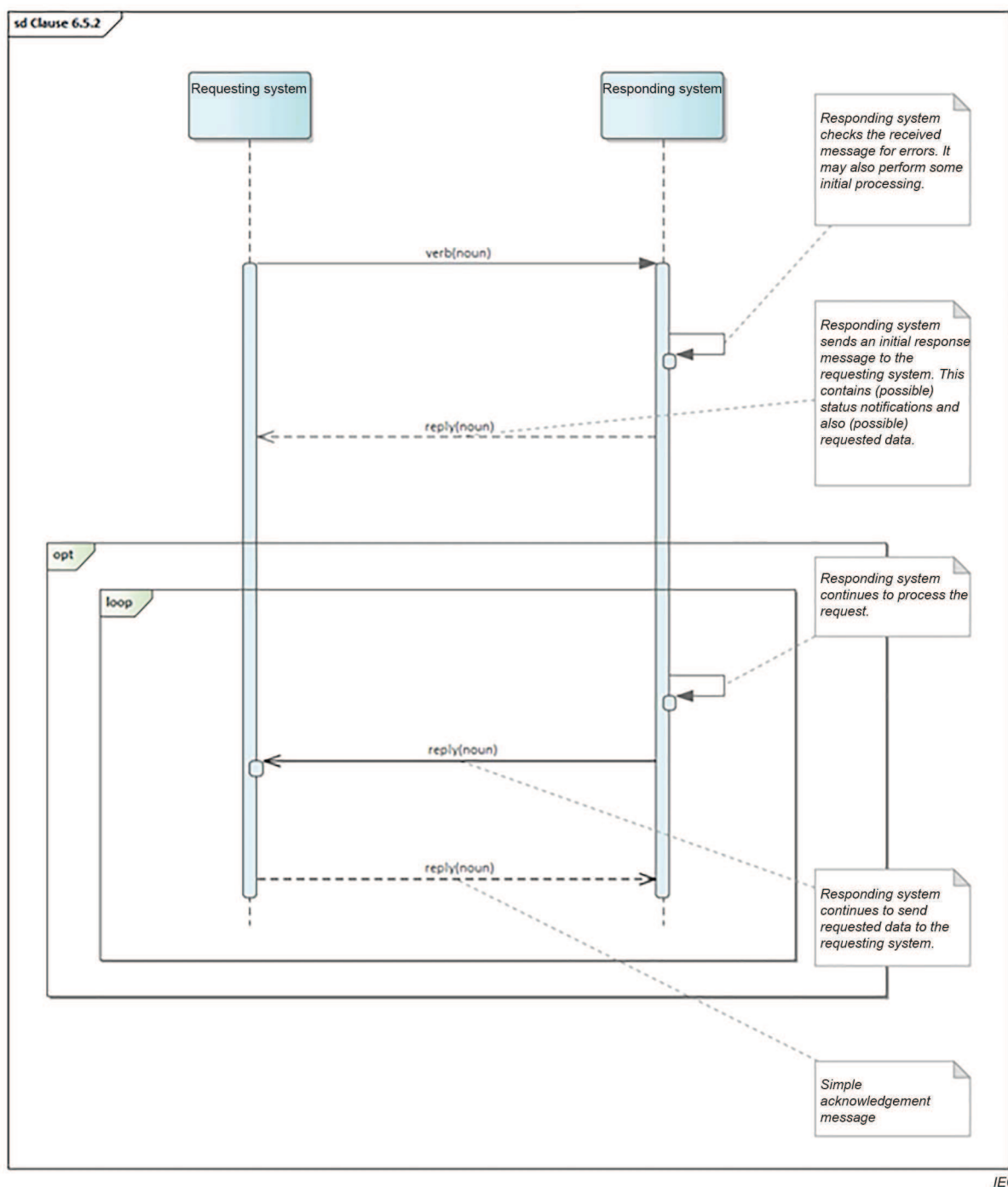


Figure 8 – Request/response message exchange pattern with simple acknowledgement messages

NOTE If the requesting system detects an error in the reply message received from the responding system, it may replace the simple acknowledgement message with a status code of 0.3 by a reply message containing a status code corresponding to the error encountered.

6.5.3 Unsolicited event message exchange pattern with simple acknowledgement message

The unsolicited event message exchange pattern of Figure 6 is modified as shown in Figure 9 when a simple acknowledgement message is to be sent.

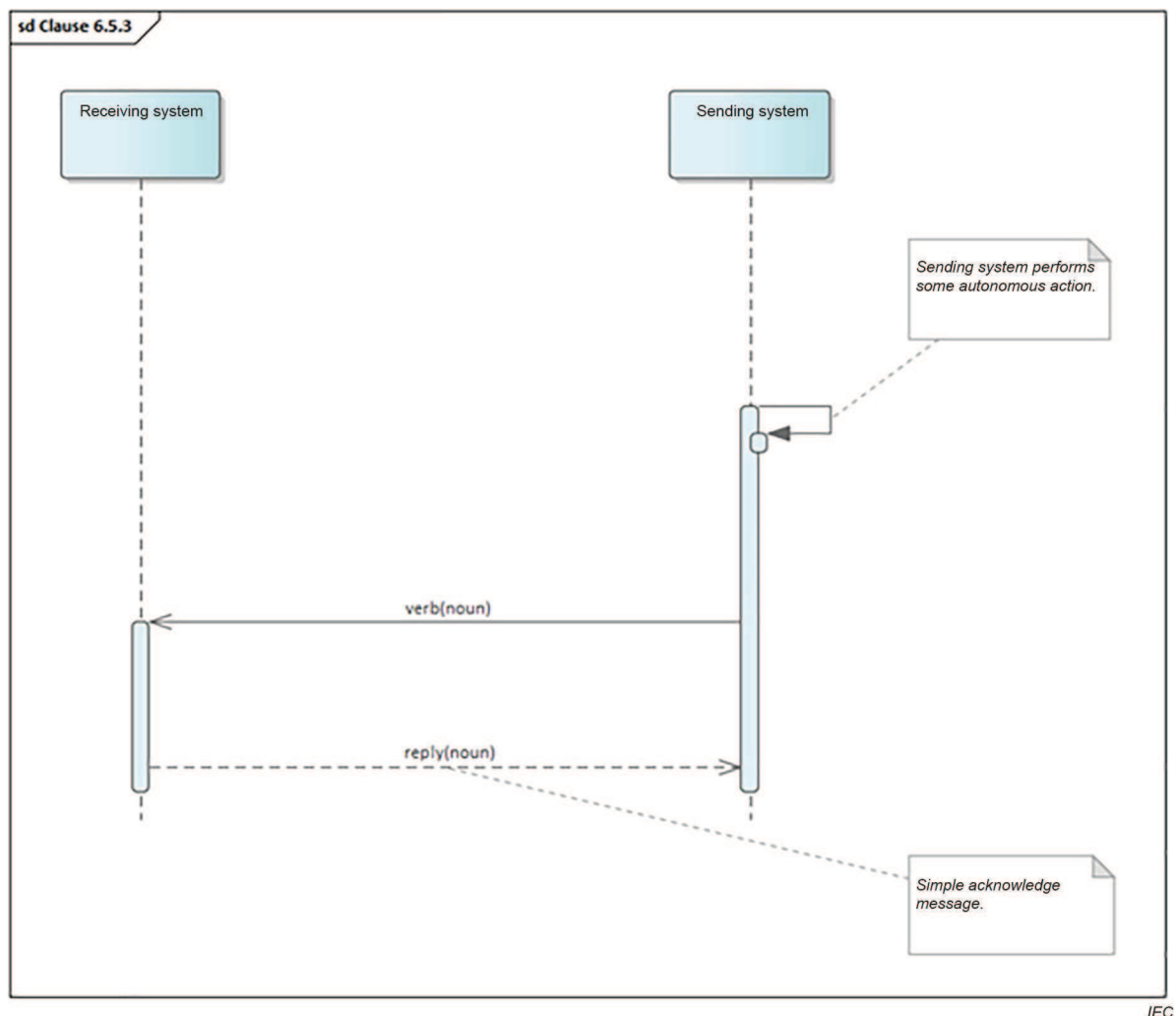


Figure 9 – Unsolicited message exchange pattern with simple acknowledgement messages

NOTE If the requesting system detects an error in the unsolicited event message received from the responding system, it may replace the simple acknowledgement message with a status code of 0.3 by a reply message containing a status code corresponding to the error encountered.

6.5.4 Consequential event message exchange pattern with simple acknowledgement messages

The consequential event message exchange pattern of Figure 7 is modified as shown in Figure 10 when simple acknowledgement messages are to be sent.

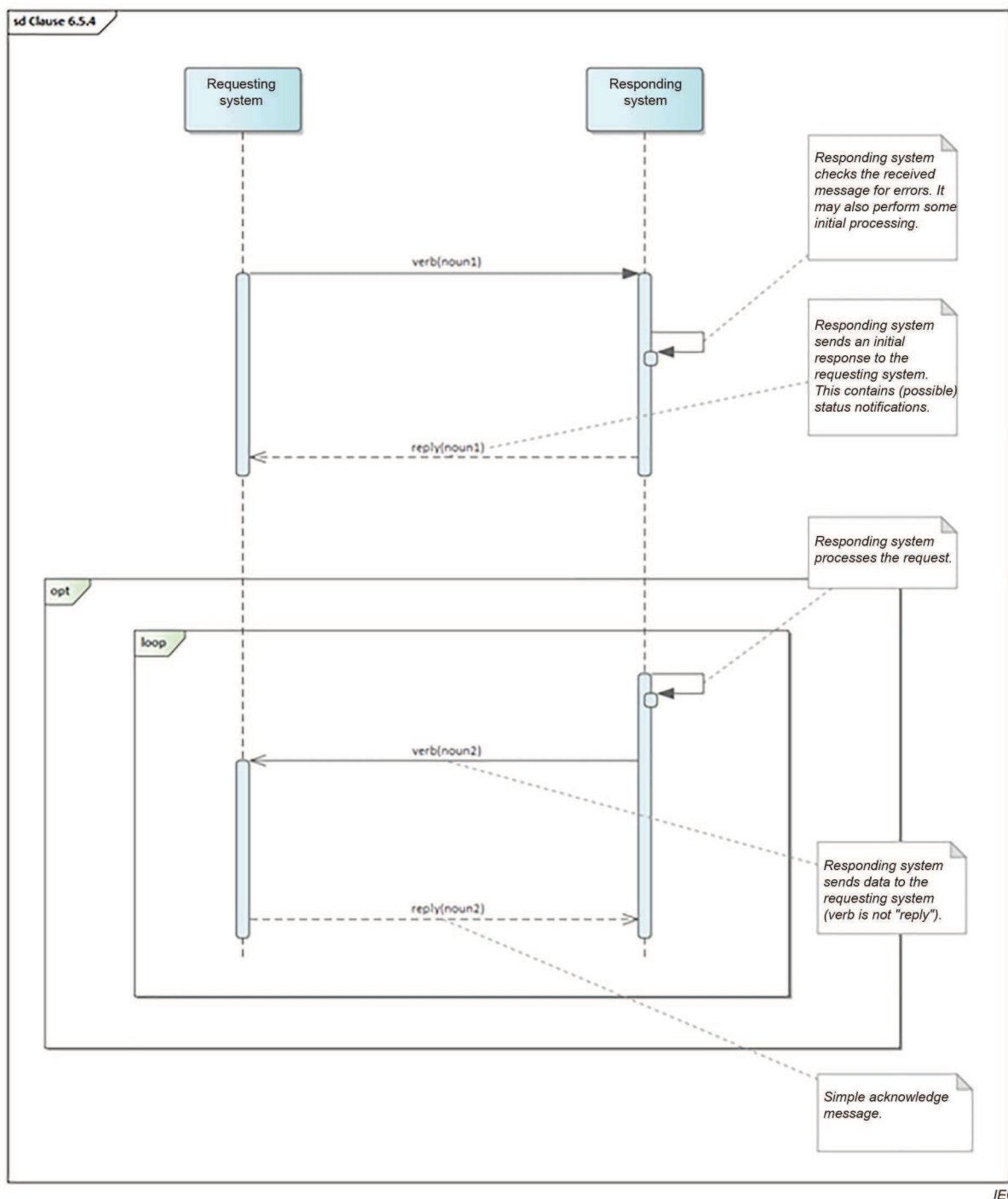


Figure 10 – Consequential event message exchange pattern with simple acknowledgement messages

NOTE If the requesting system detects an error in the consequential event message received from the responding system, it may replace the simple acknowledgement message with a status code of 0.3 by a reply message containing a status code corresponding to the error encountered.

7 Message details

7.1 General

This clause describes in detail the format of all IEC 61968-100 messages.

7.2 Message encoding

An IEC 61968-100 message shall be composed as an XML document.

7.3 Schema definition and validation

The `Message.xsd` XML schema definition file listed in Clause A.2 defines the precise syntactic structure of any IEC 61968-100 message.

An IEC 61968-100 message shall be capable of successful XML schema validation against this XSD file (Annex H).

NOTE Other standards may impose additional constraints beyond IEC 61968-100.

7.4 Top-level structure

7.4.1 General

This clause specifies the top-level structure of the IEC 61968-100 message kinds according to their respective root elements.

Other root element names shall be used in the case of strongly-typed web services (9.2.5).

7.4.2 Structure of request messages

7.4.2.1 General

This clause describes the structure of the request messages (5.2) according to their root elements.

Table 1 lists the possible message root elements and their respective `<Header><Verb>` elements (7.5.3.3).

Table 1 – List of top-level root elements in request messages

<code><Header><Verb></code> Element	Corresponding Root Element	See Clause
cancel	<code><CancelRequestMessage></code>	7.4.2.2
change	<code><ChangeRequestMessage></code>	7.4.2.3
create	<code><CreateRequestMessage></code>	7.4.2.4
delete	<code><DeleteRequestMessage></code>	7.4.2.5
execute	<code><ExecuteRequestMessage></code>	7.4.2.6
get	<code><GetRequestMessage></code>	7.4.2.7
update	<code><UpdateRequestMessage></code>	7.4.2.8

7.4.2.2 Structure of the `<CancelRequestMessage>` element

The top-level structure of a request message (5.2) whose root element is `<CancelRequestMessage>` is shown in Figure 11.

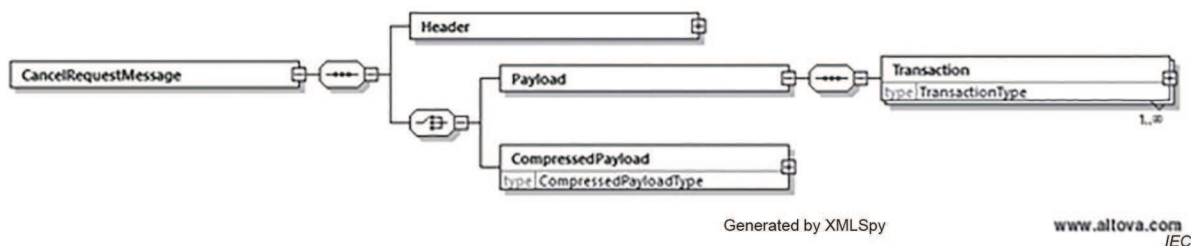


Figure 11 – Structure of the <CancelRequestMessage> element

7.4.2.3 Structure of the <ChangeRequestMessage> element

The top-level structure of a request message whose root element is <ChangeRequestMessage> is shown in Figure 12.

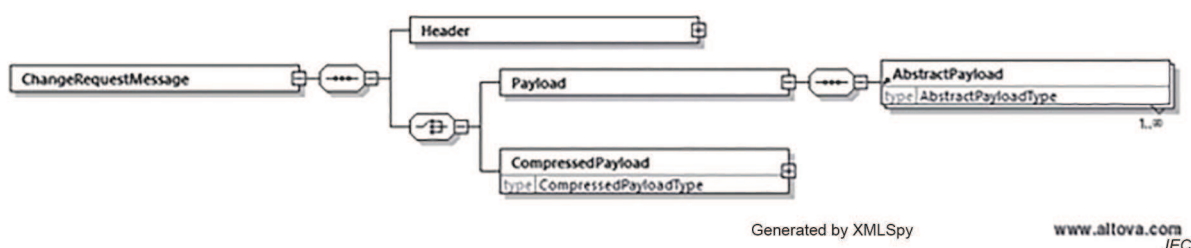


Figure 12 – Structure of the <ChangeRequestMessage> element

7.4.2.4 Structure of the <CreateRequestMessage> element

The top-level structure of a request message whose root element is <CreateRequestMessage> is shown in Figure 13.

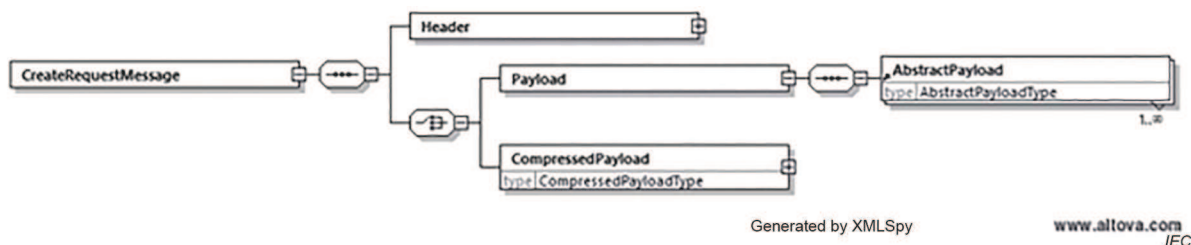


Figure 13 – Structure of the <CreateRequestMessage> element

7.4.2.5 Structure of the <DeleteRequestMessage> element

The top-level structure of the request message whose root element is <DeleteRequestMessage> is shown in Figure 14.

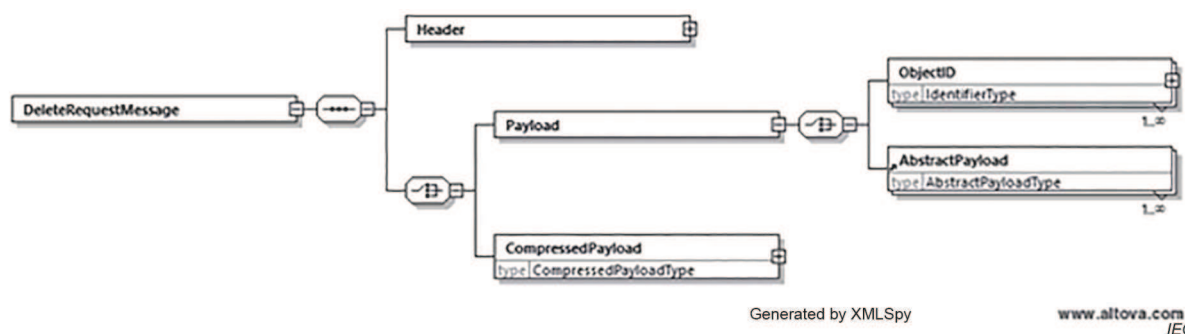


Figure 14 – Structure of the <DeleteRequestMessage> element

7.4.2.6 Structure of the <ExecuteRequestMessage> element

The top-level structure of a request message whose root element is <ExecuteRequestMessage> is shown in Figure 15.

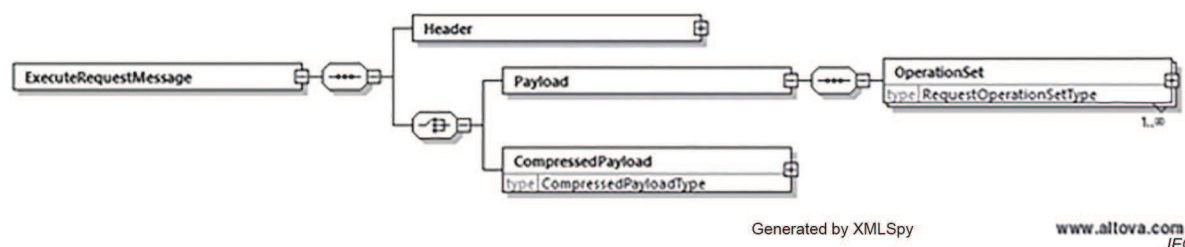


Figure 15 – Structure of the <ExecuteRequestMessage> element

7.4.2.7 Structure of the <GetRequestMessage> element

The top-level structure of a request message whose root element is <GetRequestMessage> is shown in Figure 16.

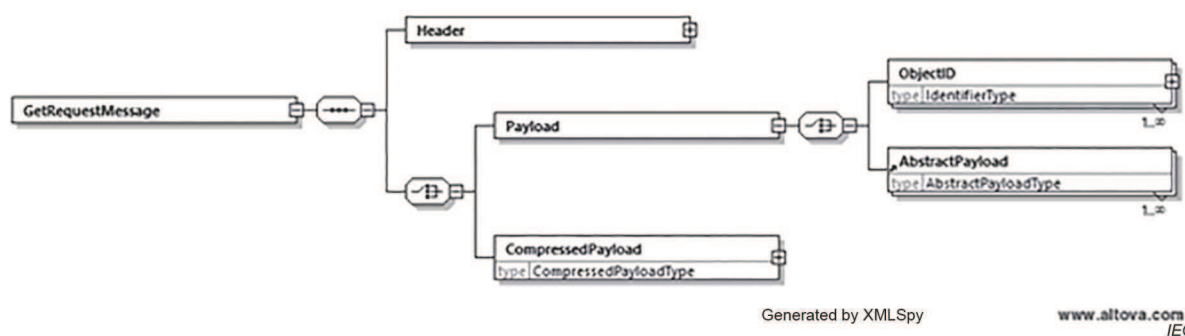


Figure 16 – Structure of the <GetRequestMessage> element

7.4.2.8 Structure of the <UpdateRequestMessage> element

The top-level structure of the request message whose root element is <UpdateRequestMessage> is shown in Figure 17.

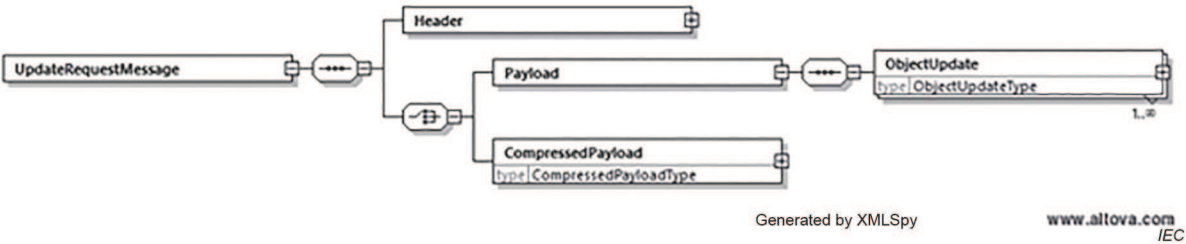


Figure 17 – Structure of the <UpdateRequestMessage> element

7.4.3 Structure of response messages

7.4.3.1 General

This clause describes the top-level structures of the response messages (5.3) according to their root elements.

Table 2 lists the possible message root elements as defined by the <Header><Verb> elements of the original request message (7.5.3.3).

Table 2 – List of top-level root elements in response messages

<Header><Verb> Element in Request Message	Corresponding Root Element	See Clause
cancel	<CancelResponseMessage>	7.4.3.2, 8.5
change	<ChangeResponseMessage>	7.4.3.3, 8.3.3
create	<CreateResponseMessage>	7.4.3.4, 8.3.2
delete	<DeleteResponseMessage>	7.4.3.5, 8.3.4
execute	<ExecuteResponseMessage>	7.4.3.6, 7.7.3
get	<GetResponseMessage>	7.4.3.7, 8.4
update	<UpdateResponseMessage>	7.4.3.8, 8.3.5

7.4.3.2 Structure of the <CancelResponseMessage> element

The structure of the response message whose root element is <CancelResponseMessage> is shown in Figure 18.

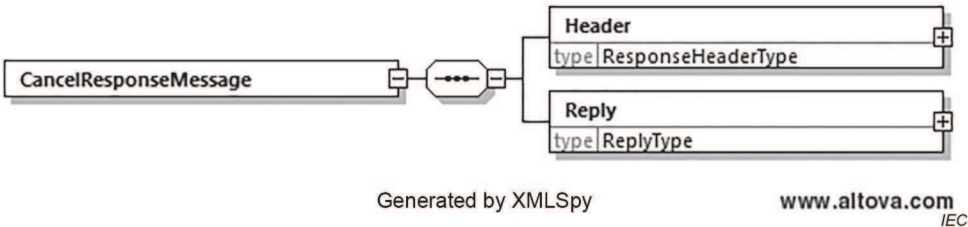


Figure 18 – Structure of the <CancelResponseMessage> element

7.4.3.3 Structure of the <ChangeResponseMessage> element

The structure of the response message whose root element is <ChangeResponseMessage> is shown in Figure 19.

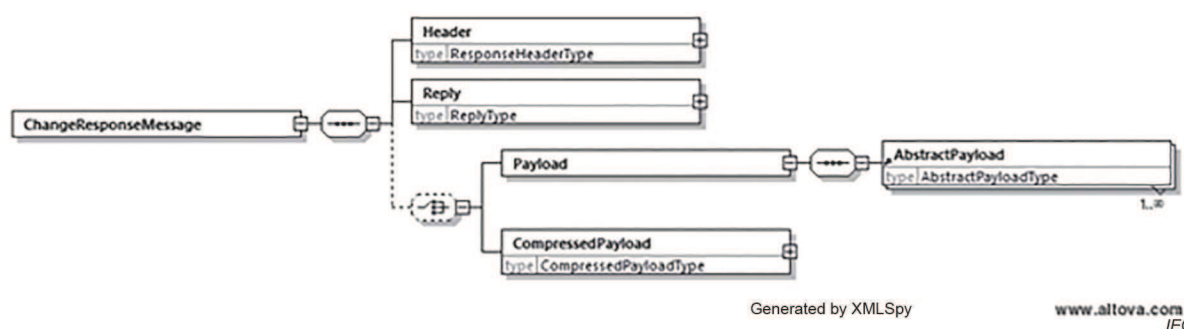


Figure 19 – Structure of the <ChangeResponseMessage> element

7.4.3.4 Structure of the <CreateResponseMessage> element

The structure of the response message whose root element is <CreateResponseMessage> is shown in Figure 20.

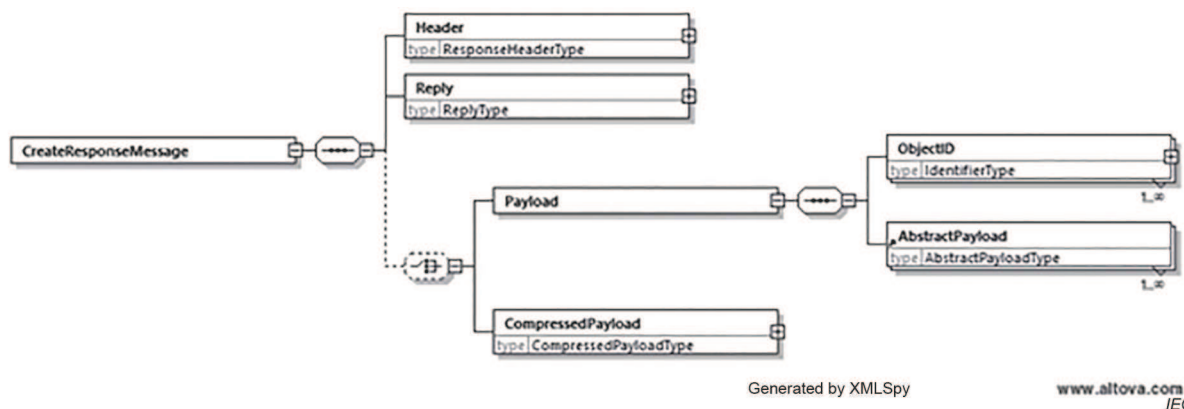


Figure 20 – Structure of the <CreateResponseMessage> element

7.4.3.5 Structure of the <DeleteResponseMessage> element

The structure of the response message whose root element is <DeleteResponseMessage> is shown in Figure 21.

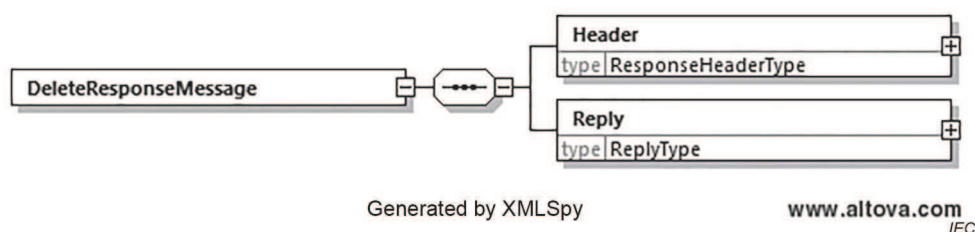


Figure 21 – Structure of the <DeleteResponseMessage> element

7.4.3.6 Structure of the <ExecuteResponseMessage> element

The structure of the response message whose root element is <ExecuteResponseMessage> is shown in Figure 22.

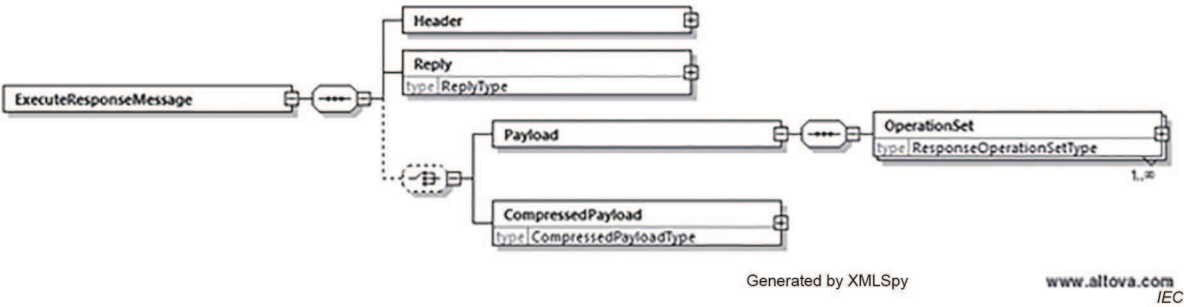


Figure 22 – Structure of the <ExecuteResponseMessage> element

7.4.3.7 Structure of the <GetReponseMessage> element

The structure of the response message whose root element is <GetResponseMessage> is shown in Figure 23.

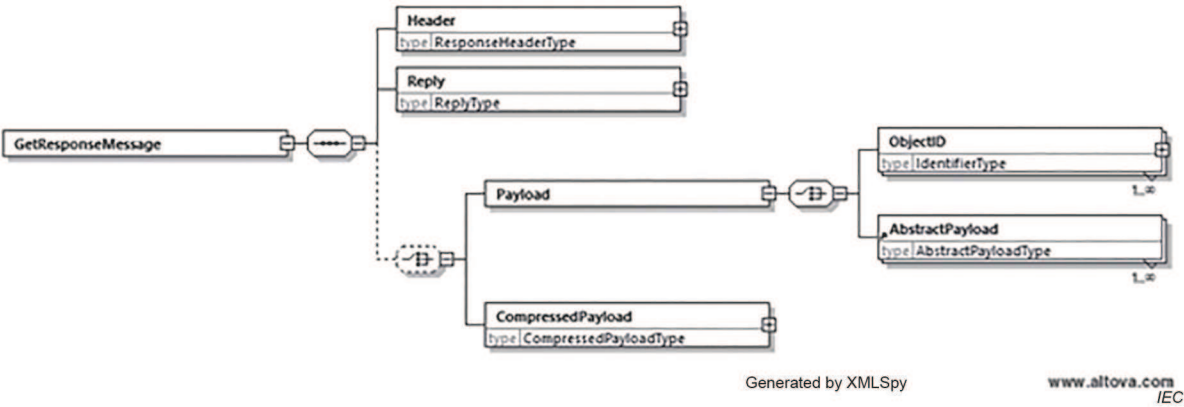


Figure 23 – Structure of the <GetResponseMessage> element

7.4.3.8 Structure of the <UpdateResponseMessage> element

The structure of the response message whose root element is <UpdateResponseMessage> is shown in Figure 24.

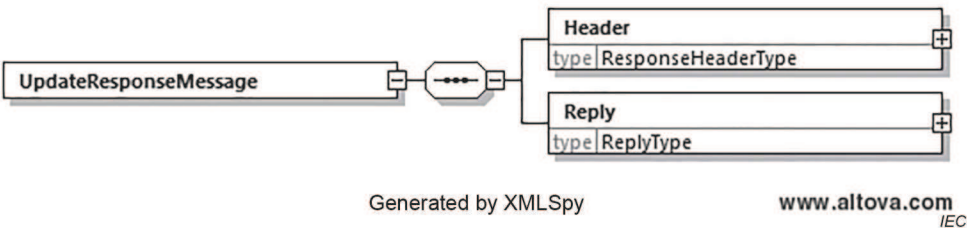


Figure 24 – Structure of the <UpdateResponseMessage> element

7.4.4 Structure of event messages

7.4.4.1 General

This clause describes the top-level structures of the event messages (5.4) according to their root elements.

Table 3 lists the possible message root elements and their respective <Header><Verb> elements (7.5.5.3)

Table 3 – List of top-level root elements in event messages

<Header><Verb> Element	Corresponding Root Element	See Clause
canceled	<CanceledEventMessage>	7.4.4.2
changed	<ChangedEventMessage>	7.4.4.3
created	<CreatedEventMessage>	7.4.4.4
deleted	<DeletedEventMessage>	7.4.4.5
executed	<ExecutedEventMessage>	7.4.4.6
updated	<UpdatedEventMessage>	7.4.4.7

7.4.4.2 Structure of the <CanceledEventMessage> element

The structure of the event message whose root element is <CanceledEventMessage> shown in Figure 25.

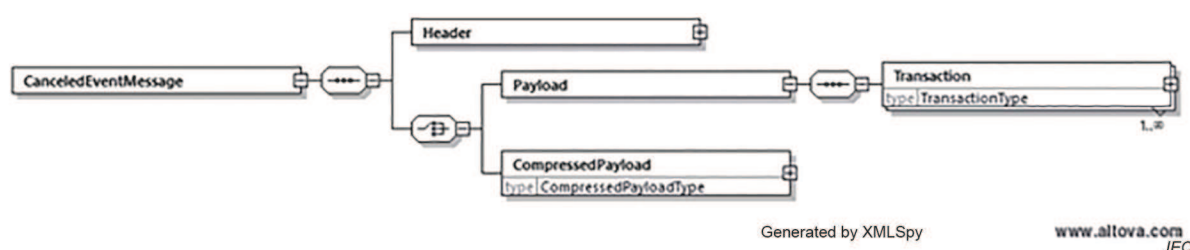


Figure 25 – Structure of the <CanceledEventMessage> element

7.4.4.3 Structure of the <ChangedEventMessage> element

The structure of the event message whose root element is <ChangedEventMessage> is shown in Figure 26.

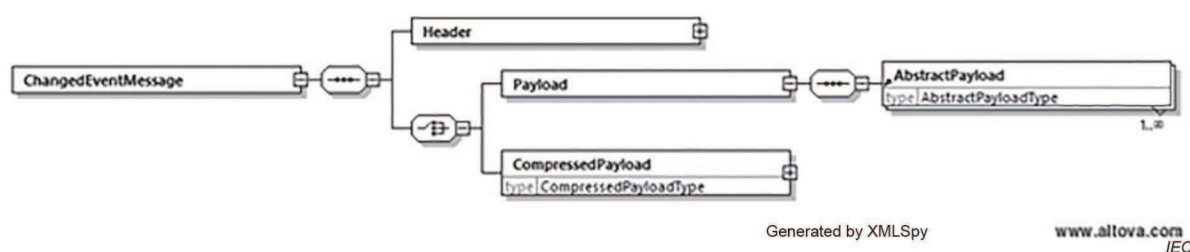


Figure 26 – Structure of the <ChangedEventMessage> element

7.4.4.4 Structure of the <CreatedEventMessage> element

The structure of the event message whose root element is <CreatedEventMessage> is shown in Figure 27.

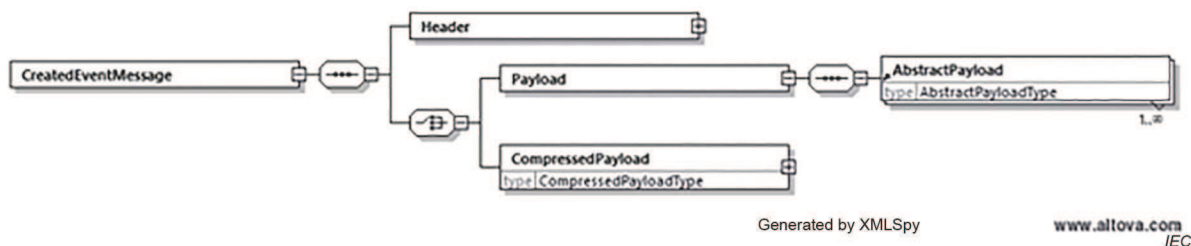


Figure 27 – Structure of the <CreatedEventMessage> element

7.4.4.5 Structure of the <DeletedEventMessage> element

The structure of the event message whose root element is <DeletedEventMessage> is shown in Figure 28.

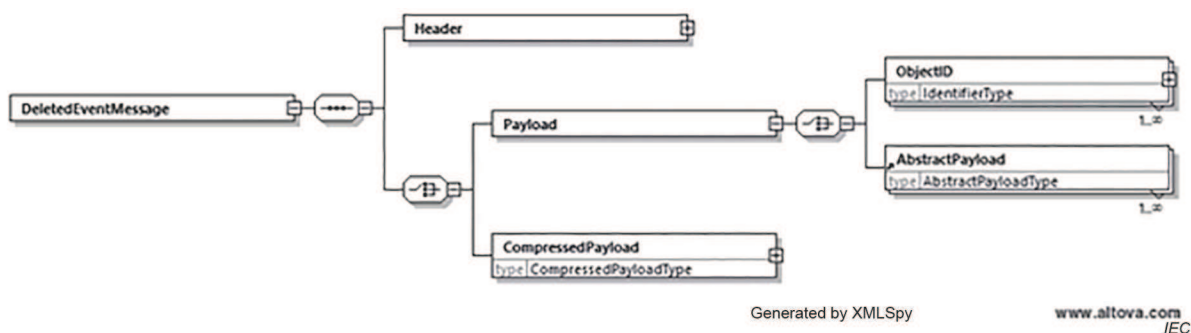


Figure 28 – Structure of the <DeletedEventMessage> element

7.4.4.6 Structure of the <ExecutedEventMessage> element

The structure of the event message whose root element is <ExecutedEventMessage> is shown in Figure 29.

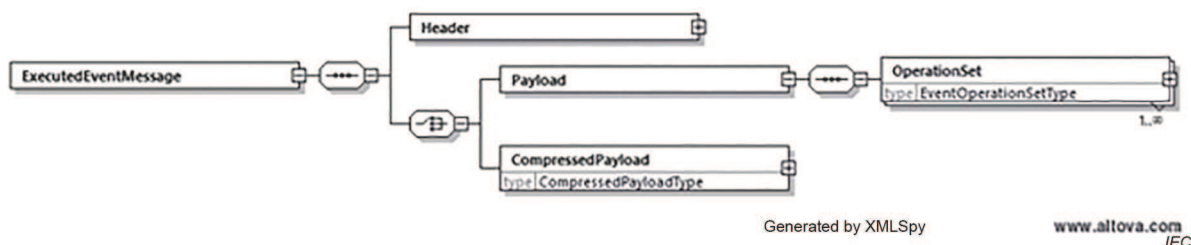


Figure 29 – Structure of the <ExecutedEventMessage> element

7.4.4.7 Structure of the <UpdatedEventMessage> element

The structure of the event message whose root element is <UpdatedEventMessage> is shown in Figure 30.

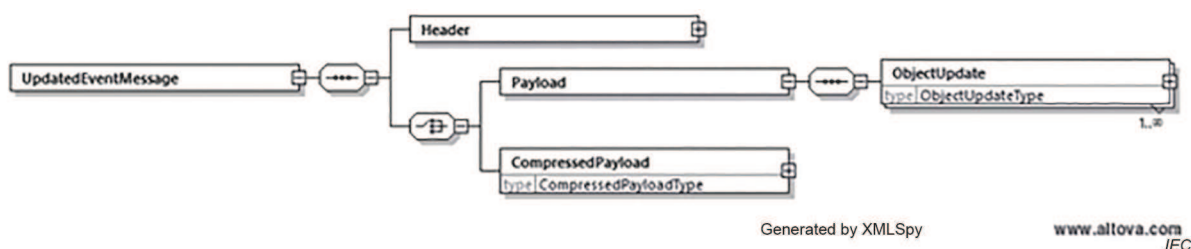


Figure 30 – Structure of the <UpdatedEventMessage> element