



Emergency Data Exchange Language (EDXL) Common Types (CT) Version 1.0

Committee Specification Draft 04

26 June 2018

Specification URIs

This version:

<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/csd04/edxl-ct-v1.0-csd04.odt> (Authoritative)
<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/csd04/edxl-ct-v1.0-csd04.html>
<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/csd04/edxl-ct-v1.0-csd04.pdf>

Previous version:

<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/csd03/edxl-ct-v1.0-csd03.odt> (Authoritative)
<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/csd03/edxl-ct-v1.0-csd03.html>
<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/csd03/edxl-ct-v1.0-csd03.pdf>

Latest version:

<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/edxl-ct-v1.0.odt> (Authoritative)
<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/edxl-ct-v1.0.html>
<http://docs.oasis-open.org/emergency/edxl-ct/v1.0/edxl-ct-v1.0.pdf>

Technical Committee:

OASIS Emergency Management TC

Chair:

Elysa Jones (elysajones@yahoo.com), Individual

Editors:

Werner Joerg (Werner.Joerg@iem.com), IEM
Rex Brooks (rexb@starbourne.com), Network Centric Operations Industry Consortium
Jeff Waters (jeff.waters@navy.mil), US Department of Defense (DoD)
Don McGarry (dmcgarry@mitre.org), MITRE Corporation

Additional artifacts:

This prose specification is one component of a Work Product that also includes:

- XML schemas: <http://docs.oasis-open.org/emergency/edxl-ct/v1.0/csd04/schema/>

Declared XML namespaces:

- <http://docs.oasis-open.org/ns/emergency/edxl-ct/v1.0>
- urn:oasis:names:tc:emergency:edxl:ct:1.0
- urn:oasis:names:tc:emergency:EDXL:CT:1.0

Abstract:

EDXL Common Types describes components and component types that can be reused across the suite of Emergency Data Exchange Language (EDXL) standards. These common components and types are intended for internal use by the Emergency Management Technical Committee and its subcommittees as they develop specific standards utilizing these types.

Status:

This document was last revised or approved by the OASIS Emergency Management TC on the above date. The level of approval is also listed above. Check the "Latest version" location noted

above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=emergency#technical.

TC members should send comments on this specification to the TC's email list. Others should send comments to the TC's public comment list, after subscribing to it by following the instructions at the "Send A Comment" button on the Technical Committee's web page at <https://www.oasis-open.org/committees/emergency/>.

This specification is provided under the [RF on RAND Terms](#) Mode of the [OASIS IPR Policy](#), the mode chosen when the Technical Committee was established. For information on whether any patents have been disclosed that may be essential to implementing this Work Product, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the TC's web page (<https://www.oasis-open.org/committees/emergency/ipr.php>).

Note that any machine-readable content ([Computer Language Definitions](#)) declared Normative for this Work Product is provided in separate plain text files. In the event of a discrepancy between any such plain text file and display content in the Work Product's prose narrative document(s), the content in the separate plain text file prevails.

Citation format:

When referencing this Work Product the following citation format should be used:

[EDXL-CT]

Emergency Data Exchange Language (EDXL) Common Types (CT) Version 1.0. Edited by Werner Joerg, Rex Brooks, Jeff Waters, and Don McGarry. 26 June 2018. OASIS Committee Specification Draft 04. <http://docs.oasis-open.org/emergency/edxl-ct/v1.0/csd04/edxl-ct-v1.0-csd04.html>. Latest version: <http://docs.oasis-open.org/emergency/edxl-ct/v1.0/edxl-ct-v1.0.html>.

Notices

Copyright © OASIS Open 2018. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](#) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <https://www.oasis-open.org/policies-guidelines/trademark> for above guidance.

Table of Contents

1	Introduction.....	5
1.1	IPR Policy.....	5
1.2	Terminology.....	5
1.3	Normative References.....	5
1.4	Non-Normative References.....	6
2	Design Principles & Concepts (non-normative).....	7
2.1	Design Philosophy.....	7
2.2	Structural Summary.....	7
2.2.1	Simple Types.....	7
2.2.2	Complex Types.....	8
2.2.3	Top Level Elements.....	9
3	EDXL Common Types Structure (normative).....	10
3.1	Data Dictionary.....	10
3.1.1	EDXL Common Simple Types.....	10
3.1.2	EDXL Enumerated Types.....	15
3.1.3	EDXL Common Complex Types.....	18
3.1.4	EDXL Common Top Level Elements.....	31
4	Conformance.....	33
A.	Acknowledgments.....	34
B.	Revision History.....	35

1 Introduction

[All text is normative unless otherwise labeled]

This document describes common components and component types that can be reused across the suite of Emergency Data Exchange Language (EDXL) standards. This document is intended for internal use by the Emergency Management Technical Committee and its subcommittees as they develop specific standards utilizing these types. The goal is to enable reuse of components which are commonly used in specifications and which have been designed based on lessons learned from the development of the Common Alert Protocol 1.1, the Distribution Element 1.0, Hospital Availability and Resource Messaging. The first use of these common components is intended to be in Situation Reports 1.0 and the Distribution Element 2.0. The components will be used and expanded as needed for future EDXL specifications.

1.1 IPR Policy

This specification is provided under the [RF on RAND Terms](#) Mode of the [OASIS IPR Policy](#), the mode chosen when the Technical Committee was established. For information on whether any patents have been disclosed that may be essential to implementing this Work Product, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the TC's web page (<https://www.oasis-open.org/committees/emergency/ipr.php>).

1.2 Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [\[RFC2119\]](#).

1.3 Normative References

[RFC2119]

S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, BCP 14, RFC 2119, March 1997. <http://www.ietf.org/rfc/rfc2119.txt>.

[RFC2046]

N. Freed, *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*, IETF RFC 2046, November 1996. <http://www.ietf.org/rfc/rfc2046.txt>.

[RFC3066]

H. Alvestrand, *Tags for the Identification of Languages*, IETF RFC 3066, January 2001. <http://www.ietf.org/rfc/rfc3066.txt>.

[WGS 84]

National Geospatial Intelligence Agency, Department of Defense, *World Geodetic System 1984*, NGA Technical Report TR8350.2, January 2000. <http://earth-info.nga.mil/GandG/publications/tr8350.2/wgs84fin.pdf>.

[XML 1.0]

T. Bray, *Extensible Markup Language (XML) 1.0 (Third Edition)*, W3C REC-XML-20040204, (Fifth Edition), 26 November 2008. <http://www.w3.org/TR/REC-xml/>.

[namespaces]

T. Bray, *Namespaces in XML*, W3C REC-xml-names-19990114, (Third Edition) W3C Recommendation 8 December 2009. <http://www.w3.org/TR/REC-xml-names/>.

[dateTime]

N. Freed, *XML Schema Part 2: Datatypes (Second Edition)*, W3C REC-xmlschema-2, October 2004. <http://www.w3.org/TR/xmlschema-2/#dateTime>.

1.4 Non-Normative References

[EDXL GFR]

EDXL General Functional Requirements, OASIS Emergency Management TC, 4 November 2004 <http://www.oasis-open.org/committees/download.php/10031/EDXL%20General%20Functional%20Requirements.doc>,

[EDXL-DE IG]

EDXL Distribution Element Implementer's Guide, Edited by Patti Aymond, 19 Aug. 2005. OASIS Working Draft WD01. http://www.oasis-open.org/committees/download.php/14120/EDXL_Implementer%27sGuide.doc

2 Design Principles & Concepts (non-normative)

2.1 Design Philosophy

Below are some of the guiding principles of the EDXL CT Specification:

1. Provide a method to capture and reuse xml types and elements for representing persons and organizations which are commonly needed across multiple EDXL standards.
2. Provide flexible mechanisms to update the EDXL CT Specification efficiently, without slowing down the EDXL standards development process.
3. Allow for easy updates to capture fixes or improvements.
4. Ease the reuse and understanding of the basic data components and protocols, important for emergency management and common across multiple specifications.
5. Speed the development of EDXL Standards through reuse of common components and thereby improve information sharing and data exchange across the local, state, tribal, national and non-governmental organizations of different professions that provide emergency response and management services.
6. Support the integration of data elements from profiles which enables efficient and effective reuse of other important open standards.

2.2 Structural Summary

About multiplicity notation: “[l..h]” designates range from lower bound l to higher bound h, with both l and h Natural numbers, $0 \leq l \leq h$, plus option “*” (unbounded) for h.

2.2.1 Simple Types

<u>Type name</u>	<u>depends on</u>
– EDXLDateTimeType	xs:dateTime
– EDXLStringType	xs:token
– PercentageType	xs:float
– ValueListURIType	xs:anyURI
– ValueType	xs:string
– CurrencyType	xs:string
– DegreesCTYPE	xs:float [-100 .. 70]
– DegreesCircleType	xs:float [0 .. 360]
– WeatherQualifierType	xs:string [enumeration]
– WeatherDescriptorType	xs:string [enumeration]
– WeatherPrecipitationType	xs:string [enumeration]

- WeatherObscurationType xs:string [*enumeration*]
- WeatherAddlPhenomType xs:string [*enumeration*]
- SkyConditionType xs:string [*enumeration*]
- RemarksType xs:string
- EstimateType xs:boolean
- LimitedString xs:string

2.2.2 Complex Types

<u>Type name</u>	<u>depends on</u>
– ValueListType	ct:valueListURI, ct:value
– ValueKeyType	ct:valueListURI, ct:value
– ValueKeyStringPairType	ct:ValueKeyType, xs:string
– ValueKeyIntPairType	ct:ValueKeyType, xs:int
– FreeTextType	ct:LimitedString, xs:string
– AlternateTextType	ct:LimitedString, xs:string
– TimePeriodType	ct:EDXLDateTimeType
– PersonTimePairType	ct:PersonDetailsType, ct:EDXLDateTimeType
– OrganizationInformationType	xpil:OrganizationDetailsType
– PersonDetailsType	xpil:PersonDetailsType
– METARType	<i>sequence of elements</i>
– stationID [1,1]	xs:string <i>restricted</i>
– observationTime [1,1]	ct:EDXLDateTimeType
– tempC [0,1]	ct:DegreesCType
– dewPointC [0,1]	ct:DegreesCType
– windDirDegrees[0,1]	ct:DegreesCircleType
– windSpeedkt [0,1]	xs:int [0 .. 300]
– windGustkt [0,1]	xs:int [0 .. 300]
– visibilityStatuteMI [0,1]	xs:float [0.0 .. 10.0]
– altimeterHP [0,1]	xs:int [800 .. 1200]
– seaLevelPressuremb [0,1]	xs:int [800 .. 1200]
– weatherPhenomenaReport [0,1]	<i>sequence of elements</i>
– qualifier[0,1]	ct:WeatherQualifierType
– descriptor [0,1]	ct:WeatherDescriptorType

- precipitation [0,1] ct:WeatherPrecipitationType
- obscuration [0,1] ct:WeatherObscurationType
- additional [0,1] ct:WeatherAddIPhenomType
- skyCondition [0,1] ct:SkyConditionType
- precip1HrIn [0,1] xs:float *restricted*
- precip3HrIn [0,1] xs:float *restricted*
- precip6HrIn [0,1] xs:float *restricted*
- precip24HrIn [0,1] xs:float *restricted*
- WeatherInfoType *sequence of elements*
 - METARString [0,1] xs:string
 - METARReadings [0,1] ct:METARType
 - weatherRemarks [0,1] xs:string
 - weatherConcerns [0,1] xs:string
- EDXLGeoPoliticalLocationType *choice of elements*
 - address [1,1] xal:AddressType
 - **xor**
 - geoCode [1,1] ct:ValueListType
- EDXLLocationType *choice of elements*
 - EDXLGeoLocation [1,1] edxl-gsf:EDXLGeoLocationType
 - **xor**
 - EDXLGeoPoliticalLocation [1,1] ct:EDXLGeoPoliticalLocationType

2.2.3 Top Level Elements

<u>Element name</u>	<u>depends on</u>
– valueListURI	ct:ValueListURIType
– value	ct:ValueType
– weatherInfo	ct:WeatherInfoType

3 EDXL Common Types Structure (normative)

3.1 Data Dictionary

Namespaces and prefixes used below include:

```
xs="http://www.w3.org/2001/XMLSchema"  
ct="urn:oasis:names:tc:emergency:edxl:ct:1.0"  
xpil="urn:oasis:names:tc:emergency:edxl:ciq:1.0:xpil"  
xal="urn:oasis:names:tc:emergency:edxl:ciq:1.0:xal"  
xnl="urn:oasis:names:tc:emergency:edxl:ciq:1.0:xnl"
```

3.1.1 EDXL Common Simple Types

Type	EDXLDateTimeType
BaseType	Restricted xs:dateTime
Restriction	Pattern "\d\d\d\d-\d\d-\d\dT\d\d:\d\d:\d\d[-,+] \d\d:\d\d"
Usage	Use wherever you would otherwise use xs:dateTime
Definition	A restricted form of dateTime which requires the use of a timezone offset and thereby prohibits the use of "Z" without an offset. Also prohibited is the use fractional seconds.
Comments	1. The uniform requirement for a timezone offset provides greater reliability and robustness for emergency systems.
Schema Component	<pre><xs:simpleType name="EDXLDateTimeType"> <xs:restriction base="xs:dateTime"> <xs:pattern value="\d\d\d\d-\d\d-\d\dT\d\d:\d\d:\d\d[-,+] \d\d:\d\d"/> </xs:restriction> </xs:simpleType></pre>
Used In	Top level type
Example	<pre><dateTimeSent>2009-11-15T16:53:00-05:00</dateTimeSent></pre>

Type	EDXLStringType
BaseType	Restricted xs:token
Restriction	minLength = 1, maxLength = 1023
Usage	Use wherever you would otherwise use xs:string
Definition	A restricted form of string which is limited to 1023 characters (and must be at least 1 character) in length
Comments	1. This common type provides a string type which is of long but limited length. Emergency systems shouldn't be required to manage indefinitely long string lengths.

	<p><code>maxLength</code> counts the maximum number of characters in the string.</p> <p>2. This type does not exclude the use of the more general <code>xs:string</code>, but should be applied whenever length limitation is technically indicated, e.g.</p> <ul style="list-style-type: none"> - to prevent circumvention of REQUIRED usage by supplying an empty string (length 0), or - for coding or transmission efficiency.
Schema Component	<pre><xs:simpleType name="EDXLStringType"> <xs:restriction base="xs:token"> <xs:maxLength value="1023"/> <xs:minLength value="1"/> </xs:restriction> </xs:simpleType></pre>
Used In	Top level type
Example	<code><senderID>mary.thompson@myagency.gov</senderID></code>

Type	LimitedString
BaseType	Restricted <code>xs:string</code>
Restriction	<code>maxLength = 1024</code> <code>whitespace = preserve</code>
Usage	Use wherever you would otherwise use <code>xs:string</code>
Definition	Text block for preserving whitespace but limiting length to 1024 characters.
Comments	<p>1. This common type provides a string type which is of long but limited length. Emergency systems shouldn't be required to manage indefinitely long string lengths. <code>maxLength</code> counts the maximum number of characters in the string.</p> <p>2. This type does not exclude the use of the more general <code>xs:string</code>, but should be applied whenever whitespace needs to be preserved for formatting and length limitation is needed.</p>
Schema Component	<pre><xs:simpleType name="LimitedString"> <xs:restriction base="xs:string"> <xs:whiteSpace value="preserve"/> <xs:maxLength value="1024"/> </xs:restriction> </xs:simpleType></pre>
Used In	Top level type
Example	

Type	PercentageType
BaseType	Restricted <code>xs:float</code>
Restriction	<code>minInclusive=0, maxInclusive=100.0</code>
Usage	Use wherever you need to use a percentage
Definition	A restricted form of unsigned floating number ranging from 0.0 to 100.0 inclusive intended to represent a percentage
Comments	1. Percentages are often used in emergency messages so this Percentage type facili-

	tates a standardized format as opposed to ad hoc percentage formats.
Schema Component	<pre><xs:simpleType name="PercentageType"> <xs:restriction base="xs:float"> <xs:minInclusive value="0"/> <xs:maxInclusive value="100.0"/> </xs:restriction> </xs:simpleType></pre>
Used In	Top level type
Example	<code><percentage>100</percentage></code>

Type	ValueListURIType
BaseType	Restricted <code>xs:anyURI</code>
Restriction	None.
Usage	Used to denote the URI of a <code>valueList</code> and related types
Definition	A URI referencing an externally-managed list of values.
Comments	1. A lesson learned from early EDXL specification development was the need to support lists of values that may vary depending on the jurisdiction or community. The <code>ValueListType</code> and related structures are based on the concept that the “list” of values can be maintained externally and referenced in the EDXL standards. The reference is handled by structures which include a <code>valueListURI</code> providing a unique identifier for the external “list” and then followed by a value or values from that list. The reason “list” is quoted is because the external structure may be an ontology or other structure adopted by the jurisdiction or community rather than just a simple list.
Schema Component	<pre><xs:simpleType name="ValueListURIType"> <xs:restriction base="xs:anyURI"/> </xs:simpleType></pre>
Used In	Top level type
Examples	

Type	ValueType
BaseType	Restricted <code>xs:string</code>
Restriction	None.
Usage	Used to denote the value(s) of a <code>valueList</code> and related types
Definition	A string value from an externally-managed list of values referenced by a <code>ValueListURI</code> .
Comments	1. A lesson learned from early EDXL specification development was the need to support lists of values that may vary depending on the jurisdiction or community. The <code>ValueListType</code> and related structures are based on the concept that the “list” of values can be maintained externally and referenced in the EDXL standards. The reference is handled by structures which include a <code>valueListURI</code> providing a unique identifier for the external “list” and then followed by a value or values from that “list”. The reason “list” is quoted is because the external structure may be an ontology or other structure adopted by the jurisdiction or community rather than just a simple list.

Schema Component	<pre><xs:simpleType name="Value"> <xs:restriction base="xs:string"/> </xs:simpleType></pre>
Used In	Top level type
Examples	

Type	RemarksType
BaseType	Restricted <code>xs:string</code>
Restriction	None.
Usage	
Definition	General comments or remarks associated with any applicable element.
Comments	<ol style="list-style-type: none"> Initially used in EDXL-SitRep SituationInformation, ManagementReportingSummary" and CasualtyAndIllnes Summary Report Types. Source: ICS 201
Schema Component	<pre><xs:simpleType name="RemarksType"> <xs:restriction base="xs:string"/> </xs:simpleType></pre>
Used In	Top level type
Examples	" Wildcat Canyon Mudslide" Disaster declared by MyCounty was initially declared by My-Township as "Pleasant Creek Neighborhood Sinkhole." Incident.

Type	EstimateType
BaseType	<code>xs:boolean</code>
Restriction	None.
Usage	
Definition	To designate whether a number or figure is actual or estimated. Values include: 'Y' = Estimated 'N' = Not Estimated.
Comments	Source: ICS 209
Schema Component	<pre><xs:simpleType name="EstimateType"> <xs:restriction base="xs:boolean"/> </xs:simpleType></pre>
Used In	Top level type
Examples	

Type	CurrencyType
-------------	---------------------

BaseType	xs:string
Restriction	Pattern "[([0-9])+[.][0-9][0-9] [A-Z][A-Z][A-Z]"
Usage	Use wherever currency is used in a specification.
Definition	A <code>CurrencyType</code> is at least one number followed by 0 or more numbers, followed by an optional fractional part, and followed by three capital letters designating the currency code (ISO 4217).
Comments	
Schema Component	<pre><xs:simpleType name="CurrencyType"> <xs:restriction base="xs:string"> <xs:pattern value="([0-9])+[.][0-9][0-9] [A-Z][A-Z][A-Z]" /> </xs:restriction> </xs:simpleType></pre>
Used In	Top level type
Examples	<pre><currency>45USD</currency></pre> <pre><currency xsi:schemaLocation="urn:oasis:names:tc:emergency:edxl:ct:1.0: EDXL_Common_Types_ver05.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema- instance" xmlns:ct="urn:oasis:names:tc:emergency:EDXL:CT:1.0">099999999999999 9999.00 AAA</currency></pre>

Type	DegreesCType
BaseType	Restricted xs:float
Restriction	minInclusive=-100.0, maxInclusive=70.0
Usage	Use wherever degree Celsius is used in a specification.
Definition	A restricted form of floating number ranging from -100.0 to 70.0 inclusive, intended to represent a temperature in degrees centigrade
Comments	
Schema Component	<pre><xs:simpleType name="DegreesCType"> <xs:restriction base="xs:float"> <xs:minInclusive value="-100.0" /> <xs:maxInclusive value="70.0" /> </xs:restriction> </xs:simpleType></pre>
Used In	METARType.tempC
Examples	<pre><tempC>37.2</tempC></pre>

Type	DegreesCircleType
BaseType	Restricted xs:float
Restriction	minInclusive= 0.0, maxInclusive=360.0
Usage	Use wherever an angle measurement in degrees is used in a specification.
Definition	A restricted form of unsigned floating number ranging from 0.0 to 360.0 inclusive, in-

	tended to represent an angle measurement in degrees.
Comments	
Schema Component	<pre><xs:simpleType name="DegreesCircleType"> <xs:restriction base="xs:float"> <xs:minInclusive value="0.0"/> <xs:maxInclusive value="360.0"/> </xs:restriction> </xs:simpleType></pre>
Used In	METARType.windDirDegrees
Examples	<windDirDegrees>32.3</windDirDegrees>

3.1.2 EDXL Enumerated Types

Type	WeatherQualifierType
BaseType	Enumeration
Values	"Light", "Moderate", "Heavy"
Usage	
Definition	A selection of qualifiers to categorize types of weather.
Comments	
Schema Component	<pre><xs:simpleType name="WeatherQualifierType"> <xs:restriction base="xs:string"> <xs:enumeration value="Light"/> <xs:enumeration value="Moderate"/> <xs:enumeration value="Heavy"/> </xs:restriction> </xs:simpleType></pre>
Used In	METARType.weatherPhenomenaReport.qualifier
Examples	<qualifier>Light</qualifier>

Type	WeatherDescriptorType
BaseType	Enumeration
Values	"Shallow", "Blowing", "Patches", "Showers", "Partial", "Drifting", "Thunderstorm", "Freezing"
Usage	
Definition	A selection of weather characteristics.
Comments	
Schema Component	<pre><xs:simpleType name="WeatherDescriptorType"> <xs:restriction base="xs:string"> <xs:enumeration value="Shallow"/> <xs:enumeration value="Blowing"/> <xs:enumeration value="Patches"/> </xs:restriction> </xs:simpleType></pre>

	<pre> <xs:enumeration value="Showers"/> <xs:enumeration value="Partial"/> <xs:enumeration value="Drifting"/> <xs:enumeration value="Thunderstorm"/> <xs:enumeration value="Freezing"/> </xs:restriction> </xs:simpleType> </pre>
Used In	METARType.weatherPhenomenaReport.descriptor
Examples	<descriptor>Showers</descriptor>

Type	WeatherPrecipitationType
BaseType	Enumeration
Values	"Drizzle", "Ice Crystals", "Unknown", "Rain", "Ice Pellets", "Snow", "Hail", "Snow Grains", "Snow Hail"
Usage	
Definition	A selection of precipitation characteristics.
Comments	
Schema Component	<pre> <xs:simpleType name="WeatherPrecipitationType"> <xs:restriction base="xs:string"> <xs:enumeration value="Drizzle"/> <xs:enumeration value="Ice Crystals"/> <xs:enumeration value="Unknown"/> <xs:enumeration value="Rain"/> <xs:enumeration value="Ice Pellets"/> <xs:enumeration value="Snow"/> <xs:enumeration value="Hail"/> <xs:enumeration value="Snow Grains"/> <xs:enumeration value="Snow Hail"/> </xs:restriction> </xs:simpleType> </pre>
Used In	METARType.weatherPhenomenaReport.precipitation
Examples	<precipitation>Drizzle</precipitation>

Type	WeatherObscurationType
BaseType	Enumeration
Values	"Mist", "Sand", "Smoke", "Haze", "Volcanic Ash", "Spray", "Wide-spread Dust", "Other"
Usage	
Definition	A selection of qualifiers to categorize types of obscuration.
Comments	
Schema Component	<pre> <xs:simpleType name="WeatherObscurationType"> <xs:restriction base="xs:string"> <xs:enumeration value="Mist"/> </pre>

	<pre> <xs:enumeration value="Sand"/> <xs:enumeration value="Smoke"/> <xs:enumeration value="Haze"/> <xs:enumeration value="Volcanic Ash"/> <xs:enumeration value="Spray"/> <xs:enumeration value="Widespread Dust"/> <xs:enumeration value="Other"/> </xs:restriction> </xs:simpleType> </pre>
Used In	METARType.weatherPhenomenaReport.obscuratation
Examples	<obscuratation>Other</obscuratation>

Type	WeatherAddlPhenomType
BaseType	Enumeration
Values	"Squall", "Funnel Cloud", "Sandstorm", "Tornado", "Waterspout", "Duststorm", "Dust Whirls"
Usage	
Definition	A selection of qualifiers for weather phenomena.
Comments	
Schema Component	<pre> <xs:simpleType name="WeatherAddlPhenomType"> <xs:restriction base="xs:string"> <xs:enumeration value="Squall"/> <xs:enumeration value="Funnel Cloud"/> <xs:enumeration value="Sandstorm"/> <xs:enumeration value="Tornado"/> <xs:enumeration value="Waterspout"/> <xs:enumeration value="Duststorm"/> <xs:enumeration value="Dust Whirls"/> </xs:restriction> </xs:simpleType> </pre>
Used In	METARType.weatherPhenomenaReport.additional
Examples	<additional>Dust Whirls</additional>

Type	SkyConditionType
BaseType	Enumeration
Values	"Sky Clear", "Few", "Scattered", "Broken", "Overcast"
Usage	
Definition	A selection of qualifiers for sky conditions.
Comments	
Schema Component	<pre> <xs:simpleType name="SkyConditionType"> <xs:restriction base="xs:string"> <xs:enumeration value="Sky Clear"/> <xs:enumeration value="Few"/> </pre>

	<pre> <xs:enumeration value="Scattered"/> <xs:enumeration value="Broken"/> <xs:enumeration value="Overcast"/> </xs:restriction> </xs:simpleType> </pre>
Used In	METARType.skyCondition
Examples	<skyCondition>Overcast</skyCondition>

3.1.3 EDXL Common Complex Types

Type	ValueListType
BaseType	xs:complexType
Restriction	None.
Usage	Use wherever a specification needs values from an externally managed list.
Definition	A ValueListType includes one valueListURI element and one or more value elements.
Comments	<ol style="list-style-type: none"> 1. A lesson learned from early EDXL specification development was the need to support lists of values that may vary depending on the jurisdiction or community. The ValueListType and related structures are based on the concept that the “list” of values can be maintained externally and referenced in the EDXL standards. The reference is handled by structures which include a valueListURI providing a unique identifier for the external “list” and then followed by a value or values from that “list”. The reason “list” is quoted is because the external structure may be an ontology or other structure adopted by the jurisdiction or community rather than just a simple list. 2. A lesson learned is that enumerated types provide a brittle, hard-to-change solution to a list of types which needs to satisfy the needs of many jurisdictions.
Schema Component	<pre> <xs:complexType name="ValueListType"> <xs:sequence> <xs:element ref="ct:valueListURI" minOccurs="1" maxOccurs="1"/> <xs:element ref="ct:value" minOccurs="1" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </pre>
Used In	Top level type
Examples	<pre> <keyword> <ct:valueListURI>urn:myagency:gov:sensors:keywords</ct:ValueListURI> <ct:value>SNM Detection</ct:value> <ct:value>XYZ</ct:value> </keyword> </pre>

Type	ValueKeyType
BaseType	xs:complexType
Restriction	None.

Usage	Use wherever a specification needs one single value from an externally managed list.
Definition	A <code>ValueKeyType</code> includes one <code>valueListURI</code> element and one and only one <code>value</code> element.
Comments	<ol style="list-style-type: none"> 1. A lesson learned from early EDXL specification development was the need to support lists of values that may vary depending on the jurisdiction or community. The <code>ValueKeyType</code> and related structures are based on the concept that the “list” of values can be maintained externally and referenced in the EDXL standards. The reference is handled by structures which include a <code>valueListURI</code> providing a unique identifier for the external “list” and then followed by a value from that “list”. The reason “list” is quoted is because the external structure may be an ontology or other structure adopted by the jurisdiction or community rather than just a simple list. 2. A lesson learned is that enumerated types provide a brittle, hard-to-change solution to a list of types which needs to satisfy the needs of many jurisdictions. 3. A lesson learned is that from some kinds of lists only one value is appropriate and multiple values would be an error. In this case, use <code>ValueKeyType</code> instead of <code>ValueListType</code>.
Schema Component	<pre><xs:complexType name="ValueKeyType"> <xs:sequence> <xs:element ref="ct:valueListURI" minOccurs="1" maxOccurs="1"/> <xs:element ref="ct:value" minOccurs="1" maxOccurs="1"/> </xs:sequence> </xs:complexType></pre>
Used In	Top level type
Examples	<pre><DistributionDefType> <ct:valueListURI> urn:oasis:names:tc:emergency:EDXL:DE:2.0:Defaults:DistributionType </ct:valueListURI> <ct:value>Report</ct:value> </DistributionDefType></pre>

Type	ValueKeyStringPairType
BaseType	<code>xs:complexType</code>
Usage	Use wherever a specification needs one single value from an externally managed list paired with a string.
Definition	A <code>ValueKeyStringPairType</code> includes one <code>valueKeyURI</code> (of type <code>ValueKeyType</code> containing a <code>valueListURI</code> and one single <code>Value</code>) followed by a <code>pairValue</code> of type <code>xs:string</code> .
Comments	
Schema Component	<pre><xs:complexType name="ValueKeyStringPairType"> <xs:sequence> <xs:element name="valueKeyURI" type="ct:ValueKeyType" minOccurs="1" maxOccurs="1"/> <xs:element name="pairValue" type="xs:string" minOccurs="1" maxOccurs="1"/> </xs:sequence> </xs:complexType></pre>
Used In	Top level type
Examples	<code><aValueKeyStringPair></code>

	<pre> <ct:valueKeyURI> <ct:valueListURI>http://example.com/lists/mylist</ct:valueListURI> <ct:value>OneStringValue</ct:value> </ct:valueKeyURI> <ct:pairValue>A Paired String</ct:pairValue> </aValueKeyStringPair> </pre>
--	--

Type	ValueKeyIntPairType
BaseType	xs:complexType
Usage	Use wherever a specification needs one single value from an externally managed list paired with an integer.
Definition	A ValueKeyIntPairType includes one valueKeyURI (of type ValueKeyType containing a valueListURI and one single value) followed by a pairValue of type xs:int.
Comments	
Schema Component	<pre> <xs:complexType name="ValueKeyIntPairType"> <xs:sequence> <xs:element name="valueKeyURI" type="ct:ValueKeyType" minOccurs="1" maxOccurs="1"/> <xs:element name="pairValue" type="xs:int" minOccurs="1" maxOccurs="1"/> </xs:sequence> </xs:complexType> </pre>
Used In	Top level type
Examples	<pre> <aValueKeyIntPair> <ct:valueKeyURI> <ct:valueListURI> http://example.com/lists/mylist </ct:valueListURI> <ct:value>OneStringValue</ct:value> </ct:valueKeyURI> <ct:pairValue>37</ct:pairValue> </aValueKeyIntPair> </pre>

Type	FreeTextType
BaseType	xs:complexType
Usage	Use wherever you would otherwise use xs:string with limited length, preserving white-space formatting.
Definition	The text value that uses the message default language (defined at in the defaultLanguage attribute)
Comments	
Schema Component	<pre> <xs:complexType name="FreeTextType"> <xs:sequence> <xs:element name="defaultText" type="ct:LimitedString"> <xs:annotation> <xs:documentation>The text value that uses the mes- sage default language (defined at in the defaultLanguage at- </pre>

	<pre> tribute).</xs:documentation> </xs:annotation> </xs:element> <xs:element name="alternateText" type="ct:AlternateText- Type" minOccurs="0" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>Alternate language representa- tion.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </pre>
Used In	Top level type
Examples	

Type	AlternateTextType
BaseType	xs:complexType
Usage	Use wherever a specified alternate language needs to be used.
Definition	Language code for the text in this element. Code MUST comply with RFC3066.
Comments	
Schema Component	<pre> <xs:complexType name="AlternateTextType"> <xs:simpleContent> <xs:extension base="ct:LimitedString"> <xs:attribute name="language" type="xs:string" use="re- quired"> <xs:annotation> <xs:documentation>Language code for the text in this element. Code MUST comply with RFC3066. </xs:documentation> </xs:annotation> </xs:attribute> </xs:extension> </xs:simpleContent> </xs:complexType> </pre>
Used In	Top level type
Examples	

Type	TimePeriodType
BaseType	xs:complexType
Usage	Use wherever a specification needs to represent a period of time.
Definition	A TimePeriodType includes one and only one fromDateTime of type ct:EDXLDateTimeType and one and only one toDateTime element of type ct:EDXLDateTimeType .
Comments	1. Time periods are commonly needed in emergency standards. This type provides a simple and useful representation of a time period which can be used for uniformity throughout the EDXL specifications.

Schema Component	<pre><xs:complexType name="TimePeriodType"> <xs:sequence> <xs:element name="fromDateTime" type="ct:EDXLDateTimeType" minOccurs="1" maxOccurs="1"/> <xs:element name="toDateTime" type="ct:EDXLDateTimeType" minOccurs="1" maxOccurs="1"/> </xs:sequence> </xs:complexType></pre>
Used In	Top level element
Examples	<pre><aTimePeriod> <ct:fromDateTime>2009-11-15T17:53:00-05:00</ct:fromDateTime> <ct:toDateTime>2009-11-15T16:53:00-05:00</ct:toDateTime> </aTimePeriod></pre>

Type	PersonTimePairType
BaseType	xs:complexType
Usage	Use wherever a specification needs to represent a person paired with a time.
Definition	A PersonTimePairType includes one and only one personDetails element of type ct:PersonDetailsType and one and only one timeValue of type ct:EDXLDateTimeType .
Comments	
Schema Component	<pre><xs:complexType name="PersonTimePairType"> <xs:sequence> <xs:element name="personDetails" type="ct:PersonDetailsType" minOccurs="1" maxOccurs="1"/> <xs:element name="timeValue" type="ct:EDXLDateTimeType" minOccurs="1" maxOccurs="1"/> </xs:sequence> </xs:complexType></pre>
Used In	Top level type
Examples	<pre><aPersonTimePair> <ct:personDetails> <xnl:PersonName> <xnl:NameElement>Mary Smith</xnl:NameElement> </xnl:PersonName> </ct:personDetails> <ct:timeValue>2009-11-15T17:53:00-05:00</ct:timeValue> </aPersonTimePair></pre>

Type	METARType
BaseType	xs:complexType
Usage	
Definition	A subset of the METAR weather data set.
Comments	This is a verbose form for reporting METAR weather information
Sub-elements	<ul style="list-style-type: none"> • stationID [1..1] of type xs:string restricted • observationTime [1..1] of type ct:EDXLDateTimeType • tempC [0..1] of type ct:DegreesCType

	<ul style="list-style-type: none"> dewPointC [0..1] of type ct:DegreesCType windDirDegrees [0..1] of type ct:DegreesCircleType windSpeedkt [0..1] of type xs:int restricted windGustkt [0..1] of type xs:int restricted visibilityStatuteMI [0..1] of type xs:float restricted altimeterHP [0..1] of type xs:int restricted seaLevelPressuremb [0..1] of type xs:float restricted weatherPhenomenaReport [0..1] of type xs:complexType skyCondition [0..1] of type ct:SkyConditionType precip1HrIn [0..1] of type xs:float restricted precip3HrIn [0..1] of type xs:float restricted precip6HrIn [0..1] of type xs:float restricted precip24HrIn [0..1] of type xs:float restricted
Schema Component	<p>See schema</p> <pre><xs:complexType name="METARType"> .. </xs:complexType></pre>
Used In	Top level type
Examples	<pre><myMETAR> <ct:stationID>KEYF</ct:stationID> <ct:observationTime>2011-04-23T01:41:00+00:00</ct:observationTime> <ct:tempC>37.2</ct:tempC> <ct:dewpointC>10.0</ct:dewpointC> <ct:windDirDegrees>32.3</ct:windDirDegrees> <ct:windSpeedkt>20</ct:windSpeedkt> <ct:windGustkt>50</ct:windGustkt> <ct:visibilityStatuteMI>1.0</ct:visibilityStatuteMI> <ct:altimeterHP>800</ct:altimeterHP> <ct:seaLevelPressuremb>800</ct:seaLevelPressuremb> <ct:weatherPhenomenaReport> ... </ct:weatherPhenomenaReport> <ct:skyCondition>Overcast</ct:skyCondition> <ct:precip1HrIn>00.01</ct:precip1HrIn> <ct:precip3HrIn>01.00</ct:precip3HrIn> <ct:precip6HrIn>01.23</ct:precip6HrIn> <ct:precip24HrIn>02.25</ct:precip24HrIn> </myMETAR></pre>

Sub-Element	[METARType.]stationID
Type	xs:string restricted
Restriction	Pattern "[A-Z]{4}"
Usage	[1..1]
Definition	Identifies the reporting station
Comments	Four-character ICAO Location Indicator
Schema Component	<pre><xs:element name="stationID" minOccurs="1"> <xs:simpleType> <xs:restriction base="xs:string"> <xs:pattern value="[A-Z]{4}"/> </xs:restriction> </xs:simpleType> </xs:element></pre>

	<pre> </xs:restriction> </xs:simpleType> </xs:element> </pre>
Used In	METARType
Examples	<ct:stationID>KEYF</ct:stationID>

Sub-Element	[METARType.]windSpeedkt
Type	xs:int restricted
Restriction	Range [0 .. 300]
Usage	[0..1]
Definition	Wind speed in knots
Comments	
Schema Component	<pre> <xs:element name="windSpeedkt" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:int"> <xs:minInclusive value="0"/> <xs:maxInclusive value="300"/> </xs:restriction> </xs:simpleType> </xs:element> </pre>
Used In	METARType
Examples	<ct:windSpeedkt>20</ct:windSpeedkt>

Sub-Element	[METARType.]windGustkt
Type	xs:int restricted
Restriction	Range [0 .. 300]
Usage	[0..1]
Definition	Speed of wind gusts in knots
Comments	
Schema Component	<pre> <xs:element name="windGustkt" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:int"> <xs:minInclusive value="0"/> <xs:maxInclusive value="300"/> </xs:restriction> </xs:simpleType> </xs:element> </pre>
Used In	METARType
Examples	<ct:windGustkt>50</ct:windGustkt>

Sub-Element	[METARType.]visibilityStatuteMI
Type	xs:float restricted
Restriction	Range [0 .. 10.0]
Usage	[0..1]
Definition	Visibility in Statute Miles
Comments	
Schema Component	<pre><xs:element name="visibilityStatuteMI" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:float"> <xs:minInclusive value="0.0"/> <xs:maxInclusive value="10.0"/> </xs:restriction> </xs:simpleType> </xs:element></pre>
Used In	METARType
Examples	<ct:visibilityStatuteMI>1.0</ct:visibilityStatuteMI>

Sub-Element	[METARType.]altimeterHP
Type	xs:int restricted
Restriction	Range [800 .. 1200]
Usage	[0..1]
Definition	Altimeter measurement in hectopascal
Comments	
Schema Component	<pre><xs:element name="altimeterHP" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:int"> <xs:minInclusive value="800"/> <xs:maxInclusive value="1200"/> </xs:restriction> </xs:simpleType> </xs:element></pre>
Used In	METARType
Examples	<ct:altimeterHP>800</ct:altimeterHP>

Sub-Element	[METARType.]seaLevelPressuremb
Type	xs:int restricted

Restriction	Range [800 .. 1200]
Usage	[0..1]
Definition	Atmospheric pressure at sea level in millibar
Comments	1 mb = 1 hPa
Schema Component	<pre><xs:element name="seaLevelPressuremb" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:int"> <xs:minInclusive value="800"/> <xs:maxInclusive value="1200"/> </xs:restriction> </xs:simpleType> </xs:element></pre>
Used In	METARType
Examples	<ct:seaLevelPressuremb>800</ct:seaLevelPressuremb>

Sub-Element	[METARType.] weatherPhenomenaReport
Type	xs:complexType
Usage	[0..1]
Definition	
Comments	
Sub-elements	<ul style="list-style-type: none"> • qualifier [0..1] of type ct:WeatherQualifierType • descriptor [0..1] of type ct:WeatherDescriptorType • precipitation [0..1] of type ct:WeatherPrecipitationType • obscuration [0..1] of type ct:WeatherObscurationType • additional [0..1] of type ct:WeatherAddlPhenomType
Schema Component	<pre><xs:element name="seaLevelPressuremb" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:int"> <xs:minInclusive value="800"/> <xs:maxInclusive value="1200"/> </xs:restriction> </xs:simpleType> </xs:element></pre>
Used In	METARType
Examples	<pre><ct:weatherPhenomenaReport> <ct:qualifier>Light</ct:qualifier> <ct:descriptor>Showers</ct:descriptor> <ct:precipitation>Drizzle</ct:precipitation> <ct:obscuration>Other</ct:obscuration> <ct:additional>Dust Whirls</ct:additional> </ct:weatherPhenomenaReport></pre>

Sub-Element	[METARType.] precip1HrIn
--------------------	--------------------------

ment	
Type	xs:float restricted
Restriction	Pattern "[0-9][0-9].[0-9][0-9]"
Usage	[0..1]
Definition	Amount of rain fall in 1 h, in inches
Comments	
Schema Component	<pre><xs:element name="precip1HrIn" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:float"> <xs:pattern value="[0-9][0-9].[0-9][0-9]" /> </xs:restriction> </xs:simpleType> </xs:element></pre>
Used In	METARType
Examples	<ct:precip1HrIn>00.01</ct:precip1HrIn>

Sub-Element	[METARType.]precip3HrIn
Type	xs:float restricted
Restriction	Pattern "[0-9][0-9].[0-9][0-9]"
Usage	[0..1]
Definition	Amount of rain fall in 3 h, in inches
Comments	
Schema Component	<pre><xs:element name="precip3HrIn" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:float"> <xs:pattern value="[0-9][0-9].[0-9][0-9]" /> </xs:restriction> </xs:simpleType> </xs:element></pre>
Used In	METARType
Examples	<ct:precip3HrIn>01.00</ct:precip3HrIn>

Sub-Element	[METARType.]precip6HrIn
Type	xs:float restricted
Restriction	Pattern "[0-9][0-9].[0-9][0-9]"
Usage	[0..1]
Definition	Amount of rain fall in 6 h, in inches
Comments	

Schema Component	<pre><xs:element name="precip6HrIn" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:float"> <xs:pattern value="[0-9][0-9].[0-9][0-9]" /> </xs:restriction> </xs:simpleType> </xs:element></pre>
Used In	METARType
Examples	<ct:precip6HrIn>01.23</ct:precip6HrIn>

Sub-Element	[METARType.]precip24HrIn
Type	xs:float restricted
Restriction	Pattern "[0-9][0-9].[0-9][0-9]"
Usage	[0..1]
Definition	Amount of rain fall in 24 h, in inches
Comments	
Schema Component	<pre><xs:element name="precip24HrIn" minOccurs="0"> <xs:simpleType> <xs:restriction base="xs:float"> <xs:pattern value="[0-9][0-9].[0-9][0-9]" /> </xs:restriction> </xs:simpleType> </xs:element></pre>
Used In	METARType
Examples	<ct:precip24HrIn>02.25</ct:precip24HrIn>

Type	WeatherInfoType
BaseType	xs:complexType
Usage	Use wherever weather info is needed in a specification.
Definition	A container to transmit predefined weather info with free format remarks and concerns
Comments	<ol style="list-style-type: none"> METAR string: raw METAR data, "the most popular format in the world for the transmission of weather data. It is highly standardized through International Civil Aviation Organization (ICAO), which allows it to be understood throughout most of the world" [Wikipedia] METAR readings: a more verbose formatted set of weather data
Sub-elements	<ul style="list-style-type: none"> METARString [0..1] of type xs:string METARReadings [0..1] of type ct:METARType weatherRemarks [0..1] of type xs:string weatherConcerns [0..1] of type xs:string
Schema Component	<pre><xs:complexType name="WeatherInfoType"> <xs:sequence> <xs:element name="METARString" type="xs:string" minOccurs="0"/> <xs:element name="METARReadings" type="ct:METARType" minOccurs="0"/> </xs:sequence> </xs:complexType></pre>

	<pre> <xs:element name="weatherRemarks" type="xs:string" minOccurs="0"/> <xs:element name="weatherConcerns" type="xs:string" minOccurs="0"/> </xs:sequence> </xs:complexType> </pre>
Used In	Top level type
Examples	<pre> <ct:weatherInfo xsi:schemaLocation="urn:oasis:names:tc:emergency:edxl:ct:1.0 EDXL_Common_Types_wd02_dpm.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ct="urn:oa- sis:names:tc:emergency:edxl:ct:1.0"> <ct:METARString>KEYF 222355Z AUTO 0000KT 4SM BR 17/17 A3022 RMK AO2 T01700170</ct:METARString> <ct:METARReadings> <ct:stationID>KEYF</ct:stationID> <ct:observationTime>2011-04-23T01:41:00+00:00</ct:observationTime> <ct:tempC>37.2</ct:tempC> <ct:dewpointC>10.0</ct:dewpointC> <ct:windDirDegrees>32.3</ct:windDirDegrees> <ct:windSpeedkt>20</ct:windSpeedkt> <ct:windGustkt>50</ct:windGustkt> <ct:visibilityStatuteMI>1.0</ct:visibilityStatuteMI> <ct:altimeterHP>800</ct:altimeterHP> <ct:seaLevelPressuremb>800</ct:seaLevelPressuremb> <ct:weatherPhenomenaReport> <ct:qualifier>Light</ct:qualifier> <ct:descriptor>Showers</ct:descriptor> <ct:precipitation>Drizzle</ct:precipitation> <ct:obscuration>Other</ct:obscuration> <ct:additional>Dust Whirls</ct:additional> </ct:weatherPhenomenaReport> <ct:skyCondition>Overcast</ct:skyCondition> <ct:precip1HrIn>00.01</ct:precip1HrIn> <ct:precip3HrIn>01.00</ct:precip3HrIn> <ct:precip6HrIn>01.23</ct:precip6HrIn> <ct:precip24HrIn>02.25</ct:precip24HrIn> </ct:METARReadings> <ct:weatherRemarks>This is weather</ct:weatherRemarks> <ct:weatherConcerns> I am concerned it may change, and that scares me... </ct:weatherConcerns> </ct:weatherInfo> </pre>

Type	OrganizationInformationType
BaseType	Extends xpil:OrganisationDetailsType
Usage	Use wherever a specification needs to specify information about an organization.
Definition	The container type for organization information elements. The OrganizationInformationType includes at least one xnl:OrganisationName and optionally Addresses, ContactNumbers, ElectronicAddressIdentifiers and OrganisationInfo. See the OASIS EM CIQ Profile for details.
Comments	1. Note that some elements use the American spelling "Organization" and some the English spelling "Organisation".
Schema Component	<pre> <xs:complexType name="OrganizationInformationType"> <xs:complexContent> <xs:extension base="xpil:OrganisationDetailsType"/> </xs:complexContent> </pre>

	</xs:complexType>
Used In	Top level type
Examples	<pre><anOrganizationInformation> <xnl:OrganisationName> <xnl:NameElement>Corporation XYZ</xnl:NameElement> </xnl:OrganisationName> </anOrganizationInformation></pre>

Type	PersonDetailsType
Type	xpil:PersonDetailsType
Usage	Used in the PersonTimePairType.
Definition	A container for defining the unique characteristics of a person only. PersonDetailsType is an extension of xpil:PersonDetailsType which is defined in the OASIS EM TC CIQ profile xpil schema to include at least one PersonName, and optionally one Addresses, ContactNumbers, ElectronicAddressIdentifiers and Identifiers. For more information, see the OASIS EM TC CIQ profile.
Comments	1. See the EM-TC CIQ Profile
Schema Component	<pre><xs:complexType name="PersonDetailsType"> <xs:complexContent> <xs:extension base="xpil:PersonDetailsType"/> </xs:complexContent> </xs:complexType></pre>
Used In	PersonTimePairType
Examples	<pre><aPersonDetails> <ct:personDetails> <xnl:PersonName> <xnl:NameElement>Mary Smith</xnl:NameElement> </xnl:PersonName> </ct:personDetails> </aPersonDetails></pre>

Type	EDXLGeoPoliticalLocationType
BaseType	xs:complexType
Restriction	Choice.
Usage	Use wherever a specification needs a geopolitical location described as address or by geo-code.
Definition	A container for defining Geo-Political Locations.
Comments	
Schema Component	<pre><xs:complexType name="EDXLGeoPoliticalLocationType"> <xs:choice> <xs:element name="address" type="xal:AddressType"/> <xs:element name="geoCode" type="ct:ValueListType"/> </xs:choice> </xs:complexType></pre>

Used In	Top level type
Examples	

Type	EDXLLocationType
BaseType	xs:complexType
Restriction	Choice.
Usage	Use wherever a specification needs a designation of a location.
Definition	A Container for describing both Geo-Political and Geographic Locations.
Comments	
Schema Component	<pre><xs:complexType name="EDXLLocationType"> <xs:choice> <xs:element name="EDXLGeoLocation" type="edxl-gsf:EDXLGeoLocationType"/> <xs:element name="EDXLGeoPoliticalLocation" type="ct:EDXLGeoPoliticalLocationType"/> </xs:choice> </xs:complexType></pre>
Used In	Top level type
Examples	

3.1.4 EDXL Common Top Level Elements

Element	valueListURI
Type	ct:ValueListURIType
Usage	Used to denote the URI of a ValueListType and related types
Definition	A URI referencing an externally-managed list of values.
Comments	<p>1. A lesson learned from early EDXL specification development was the need to support lists of values that may vary depending on the jurisdiction or community. The ValueListType and related structures are based on the concept that the “list” of values can be maintained externally and referenced in the EDXL standards. The reference is handled by structures which include a ValueListURI providing a unique identifier for the external “list” and then followed by a value or values from that list. The reason “list” is quoted is because the external structure may be an ontology or other structure adopted by the jurisdiction or community rather than just a simple list.</p>
Schema Component	<pre><xs:element name="valueListURI" type="ValueListURIType"/></pre>
Used In	ValueListType ValueKeyType ValueKeyStringPairType ValueKeyIntPairType
Examples	<pre><ct:valueListURI>http://example.com/mylist</ct:valueListURI> <ct:valueListURI></pre>

	urn:oasis:names:tc:emergency:edxl:de:2.0:Defaults:DistributionType </ct:valueListURI>
--	--

Element	value
Type	ct:ValueType
Usage	Used to denote the value(s) of a <code>ValueListType</code> and related types
Definition	A string value from an externally-managed list of values referenced by a <code>ValueListURI</code> .
Comments	1. A lesson learned from early EDXL specification development was the need to support lists of values that may vary depending on the jurisdiction or community. The <code>ValueListType</code> and related structures are based on the concept that the “list” of values can be maintained externally and referenced in the EDXL standards. The reference is handled by structures which include a <code>ValueListURI</code> providing a unique identifier for the external “list” and then followed by a value or values from that “list”. The reason “list” is quoted is because the external structure may be an ontology or other structure adopted by the jurisdiction or community rather than just a simple list.
Schema Component	<xs:element name="value" type="ValueType"/>
Used In	<code>ValueListType</code> <code>ValueKeyType</code> <code>ValueKeyStringPairType</code> <code>ValueKeyIntPairType</code>
Examples	<ct:value>SomeValue</ct:value>

4 Conformance

The last numbered section in the specification must be the Conformance section. Conformance Statements/Clauses go here.

TBD

A. Acknowledgments

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Participants:

Don McGarry, MITRE Corp., Member

Jeff Waters, DoD, Member

Rex Brooks, Network Centric Operations Industry Consortium (NCOIC), Member

Werner Joerg, Individual, Member

B. Revision History

Revision	Date	Editor	Changes Made
WD01	03/02/2011	Jeff Waters	Initial Setup
WD02	04/21/2011	Werner Joerg	Adaptation to new schema; ready for TC review
WD03	05/02/2011	Werner Joerg	Expanded WeatherInfo; ready for TC review
	05/10/2011	Werner Joerg	Fixed link for [WGS 84] reference
WD04	11/15/2011	Werner Joerg	Added RemarksType, EstimateType, EDXLGeoPoliticalLocationType and EDXLLocationType; fixed statements in 2.1
WD05	07/01/2014 07/09/2014 07/23/2014	Werner Joerg	Update for compliance with EDXL Naming and Capitalization Guidelines (EM-TC 12/18/2012)
WD06	12/31/14	Werner Joerg	Updated formatting of references
WD07	06/12/18	Rex Brooks	Updated Imported Schema, Added FreeTextType, AlternateTextType and LimitedStringType