ETSI TS 103 221-1 V1.23.1 (2026-03)

TECHNICAL SPECIFICATION

Lawful Interception (LI);
Internal Network Interfaces;
Part 1: X1

Reference
RTS/LI-00310-1

Keywords
interface, lawful interception

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI IPR online database.

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM®** and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Lawful Interception (LI).

The present document is part 1 of a multi-part deliverable covering the Internal Network Interfaces for Lawful Interception (LI), as identified below:

Part 1: "X1";

Part 2: "X2/X3".

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document defines an electronic interface for the exchange of information relating to the establishment and management of Lawful Interception. Typically, this interface would be used between a central LI administration function and the network internal interception points.

Typical reference models for LI define an interface between Law Enforcement Agencies (LEAs) and Communication Service Providers (CSPs), called the handover interface. They also define an internal network interface within the CSP domain between administration and mediation functions for lawful interception and network internal functions, which facilitates the interception of communication. This internal network interface typically consists of several sub-interfaces: initial configuration of the network internal elements of lawful interception (X0), administration (X1), transmission of intercept related information (X2) and transmission of content of communication (X3). The present document specifies the administration interface X1.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found in the ETSI docbox.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document.

[1] ETSI TS 133 107: "Universal Mobile Telecommunications System (UMTS); LTE; Digital cellular telecommunications system (Phase 2+) (GSM); 3G security; Lawful interception architecture and functions (3GPP TS 33.107)".

[2] IETF RFC 4122: "A Universally Unique IDentifier (UUID) URN Namespace", (July 2005).

[3] W3C® Recommendation 28 October 2004: "XML Schema Part 2: Datatypes Second Edition".

[4] ETSI TS 103 280: "Lawful Interception (LI); Dictionary for common parameters".

[5] Recommendation ITU-T E.212: "The international identification plan for public networks and subscriptions".

[6] ETSI TS 123 003: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; 5G; Numbering, addressing and identification (3GPP TS 23.003)".

[7] IETF RFC 3261: "SIP: Session Initiation Protocol", (June 2002).

[8] IETF RFC 3966: "The tel URI for Telephone Numbers", (December 2004).

[9] IETF RFC 3508: "H.323 Uniform Resource Locator (URL) Scheme Registration", (April 2003).

[10] IETF RFC 7542: "The Network Access Identifier", (May 2015).

[11] IETF RFC 2865: "Remote Authentication Dial In User Service (RADIUS)", (June 2000).

[12] IETF RFC 2818: "HTTP over TLS", (May 2000).

[13] IETF RFC 7230: "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", (June 2014).

NOTE: Obsoleted by IETF RFC 9110, IETF RFC 9112.

[14] IETF RFC 5246: "The Transport Layer Security (TLS) Protocol Version 1.2", (August 2008).

NOTE: Obsoleted by IETF RFC 8446.

[15] Void.

[16] IETF RFC 7525: "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", (May 2015).

NOTE: Obsoleted by IETF RFC 9325.

[17] IETF RFC 6125: "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", (March 2011).

[18] IETF RFC 4519: "Lightweight Directory Access Protocol (LDAP): Schema for User Applications", (June 2006).

[19] ETSI TS 103 221-2: "Lawful Interception (LI); Internal Network Interfaces; Part 2: X2/X3".

[20] IETF RFC 8446: "The Transport Layer Security (TLS) Protocol Version 1.3", (August 2018).

[21] IETF RFC 7540: "Hypertext Transfer Protocol Version 2 (HTTP/2)", (May 2015).

NOTE: Obsoleted by IETF RFC 9113.

[22] ETSI TS 133 127: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; 5G; Lawful Interception (LI) architecture and functions (3GPP TS 33.127)".

[23] IETF RFC 6530: "Overview and Framework for Internationalized Email", (February 2012).

[24] W3C® Recommendation 21 March 2017: "XPath and XQuery Functions and Operators 3.1".

[25] IETF RFC 6920: "Naming Things with Hashes", (April 2013).

[26] FIPS PUB 202: "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions".

[27] IETF RFC 7042: "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", (October 2013).

[28] ETSI TS 103 120: "Lawful Interception (LI); Interface for warrant information".

[29] IETF RFC 5280: "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", (May 2008).

[30] ETSI TS 104 000: "Lawful Interception (LI); Internal Network Interface X0".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents may be useful in implementing an ETSI deliverable or add to the reader's understanding, but are not required for conformance to the present document.

[i.1] OWASP: "Transport Layer Security Cheat Sheet".

[i.2] ETSI TR 103 308: "CYBER; Security baseline regarding LI and RD for NFV and related platforms".

[i.3] ETSI GS NFV-SEC 009: "Network Functions Virtualisation (NFV); NFV Security; Report on use cases and technical approaches for multi-layer host administration".

[i.4] ETSI GS NFV-SEC 012: "Network Functions Virtualisation (NFV) Release 3; Security; System architecture specification for execution of sensitive NFV components".

[i.5] OWASP: "XML Security Cheat Sheet".

[i.6] GSMA RCC.07: "Rich Communication Suite - Advanced Communications Services and Client Specification".

[i.7] ETSI TS 102 232-1: "Lawful Interception (LI); Handover Interface and Service-Specific Details (SSD) for IP delivery; Part 1: Handover specification for IP delivery".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

destination: point to which xIRI and/or xCC is delivered by the NE

Destination IDentifier (DID): identifier to uniquely identify a Destination internally to the X1 interface

Destination Set: collection of DIDs and their associated preference of use

Destination Set IDentifier (DSID): identifier to uniquely identify a Destination Set internally to the X1 interface

Network Element (NE): element performing the LI operations such as interception, or mediation and delivery

NOTE: The NE may be embedded in an NF or standalone.

Network Function (NF): function that contains an associated or embedded NE

protocol error: error at the X1 protocol level (rather than any fault with ADMF or NE)

NOTE: In the present document, the term "error" in general refers to a protocol error, whereas issues with systems not behaving correctly are called "faults".

task: continuous instance of interception at a single NE carried out against a set of target identifiers, identified by an X1 Identifier, starting from an activate command and ending with a deactivate command or terminating fault

terminating fault: fault signalled from NE to ADMF which terminates the specific Task

X1: LI interfaces internal to the CSP for management tasking

X1 Context: portion of Controlled Function ("NE") state associated with the X1 operations controlled by a specific Controlling Function ("ADMF")

NOTE: When multiple ADMFs operate on an NE, the NE maintains a separate independent X1 Context for each of the ADMFs. System-wide, a X1 Context is uniquely identified by a combination of ADMF ID and NE ID.

X1 IDentifier (XID): identifier to uniquely identify a Task internally to the X1 interface as well as across related X2 and X3 interfaces

NOTE: The XID is also either associated to only one LIID or can be allowed to be associated to multiple LIIDs.

X1 Transaction ID: identifier used to identify a specific request/response pair

X2: LI interfaces internal to the CSP for xIRI delivery

X3: LI interfaces internal to the CSP for xCC delivery

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ADMF	ADMinistration Function
AVP	Attribute-Value Pair
AXRI	Additional XID Related Information
CC	Content of Communication
CIDR	Classless Inter Domain Routing
CIN	Communication Identity Number
CSP	Communication Service Provider
DID	Destination IDentifier
DSID	Destination Set IDentifier
EUI	Extended Unique Identifier
FQDN	Full Qualified Domain Name
GTP-C	GPRS Tunnel Protocol (Control plane)
GTP-U	GPRS Tunnel Protocol (User plane)
HI	Handover Interface
HI1	Handover Interface 1 (for administrative information)
HI2	Handover Interface 2 (for IRI)
HI3	Handover Interface 3 (for CC)
HTTP	HyperText Transfer Protocol
HTTPS	HTTP over TLS
IMEI	International Mobile Equipment Identity
IMEISV	International Mobile Equipment Identity Software Version
IMPI	IP Multimedia Private Identity
IMPU	IP Multimedia PUblic identity
IMSI	International Mobile Station Identity
IP	Internet Protocol
IRI	Intercept Related Information
LEA	Law Enforcement Agency
LEMF	Law Enforcement Monitoring Facility
LI	Lawful Interception
LIID	Lawful Interception IDentifier
MAC	Media Access Control
MDF	Mediation and Delivery Function
MSISDN	Mobile Station International Subscriber Directory Number
NAI	Network Access Identifier
NAT	Network Address Translation
NE	Network Element

NOTE: The element or function performing the interception.

NF	Network Function
NFV	Network Functions Virtualisation
OID	Object ID
OWASP	Open Web Application Security Project
POI	Point Of Interception
QoS	Quality of Service
RADIUS	Remote Authentication Dial In User Service
RCS	Rich Communication Suite
RDN	Relative Distinguished Name
SAN	Subject Alternative Name
SGSN	Serving GPRS Support Node

SIP	Session Initiation Protocol
SIP-URI	Session Initiation Protocol Uniform Resource Identifier
SNMP	Simple Network Management Protocol
SUCI	SUbscription Concealed Identifier
TCP	Transmission Control Protocol
TEL-URI	Telephony Uniform Resource Identifier
TF	Triggering Function
TISPAN	Telecommunication and Internet converged Services and Protocols for Advanced Networking
TLS	Transport Layer Security
TPM	Trusted Platform Module
UDP	User Datagram Protocol
UID	User IDentifier
URI	Uniform Resource Identifier
URN	Uniform Resource Name
UTF	UCS Transformation Formats
UUID	Universally Unique Identifier
VRF	Virtual Routing and Forwarding
xCC	X3 Content of Communications
XID	X1 IDentifier
xIRI	X2 Intercept Related Information
XML	eXtended Markup Language
XSD	XML Schema Definition

4 Overview

4.1 Reference model

4.1.1 Overview

The X1 interface is based on communication between two entities; the controlling function (e.g. a CSP ADMinistration Function (ADMF)), and the controlled function (e.g. a Network Element performing interception or mediation and delivery). The X1 reference model is shown in figure 4.1.1-1.

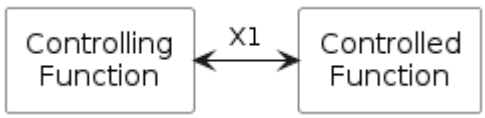

Figure 4.1.1-1: X1 reference model

The X1 model supports "many-to-many" cardinality of the communicating entities. When multiple Controlling Functions operate on a single Controlled Function, the Controlled Function maintains a separate context, called X1 Context, for each Controlling Function it communicates with. All operations and information exchanges related to a specific Controlling Function are executed within the respective X1 Context at the Controlled Function. A Controlling Function shall not be able to determine the existence or contents of X1 Contexts belonging to other Controlling Functions at a Controlled Function via X1.

In the present document the terms "NE" and "ADMF" are used both as respective equivalents to the terms "Controlled Function" and "Controlling Function", and as references to the actual LI network deployment entities. In the latter case, the term Network Element (NE) represents an element of any given Network Function (NF) which performs lawful interception. The NE is given information regarding interception or mediation and delivery. Similarly, the term "ADMF" represents the CSP's LI Administration Function that controls interception or mediation and delivery in NEs.

4.1.2 ADMF deployment model

Figure 4.1.2-1 shows a deployment model for X1 where a CSP ADMF uses X1 to provision a number of NEs to perform interception.

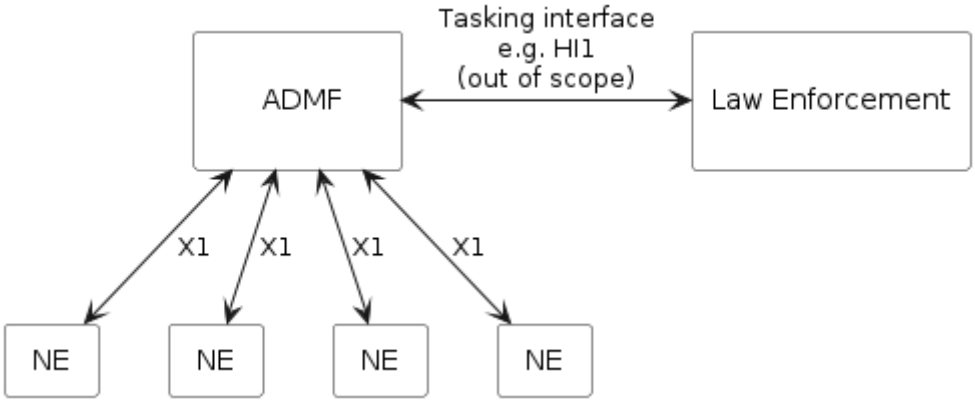

Figure 4.1.2-1: X1 Model for CSP ADMF Deployment

Onward delivery of information from the NE is called X2 (for xIRI) and X3 (for xCC). X2 and X3 are defined in ETSI TS 103 221-2 [19].

While in a typical CSP deployment there is only one ADMF, some deployments may involve multiple ADMFs for redundancy or other purposes.

ADMF and NE shall implement time synchronization where possible; in situations where it is not possible, the ADMF shall maintain knowledge of the timing offset between the ADMF and NE.

NOTE: The present document may be used in direct delivery scenarios, in which the NE delivers directly to the LEMF. Any consequences of using direct delivery are out of scope of the present document.

4.1.3 Triggering deployment model

Figure 4.1.3-1 shows another possible deployment model for X1, where the X1 protocol is used to trigger interception in an NE present in a different network function. In this deployment model, the "Triggering Function" (TF) takes on the role of the ADMF in the previous deployment model, while the "Triggered Function" takes on the role of the NE.

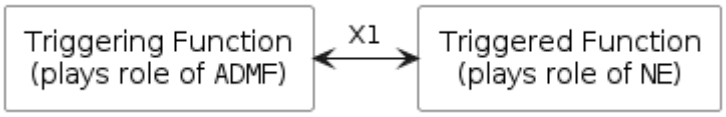

Figure 4.1.3-1: X1 deployment model for Triggering Functions

If this deployment model is used, then in the following clauses references to the ADMF should be interpreted as applying to the Triggering Function, while references to the NE should be interpreted as references to the Triggered Function.

4.1.4 Mediation and delivery function deployment model

Figure 4.1.4-1 shows another possible deployment model for X1, where the X1 protocol is used to manage a CSP mediation and delivery function. In this deployment model, the MDF takes on the role of the NE in the previous deployment model.

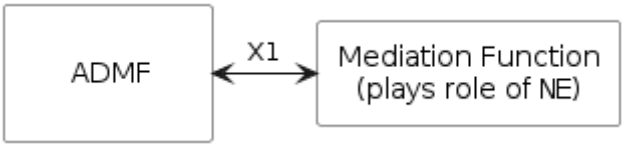

Figure 4.1.4-1: X1 deployment model for Mediation and Delivery Functions

If this deployment model is used, then in the following clauses references to the NE should be interpreted as applying to the MDF.

4.2 Reference model for X1: requesting and responding

X1 transactions consist of a request followed by a response.

Requests may be sent in either direction i.e. with the ADMF or NE initiating the request. The side initiating the request is called the "Requester"; this term is used when it is not specified whether it is the ADMF or NE making the request. The other side is called the "Responder".

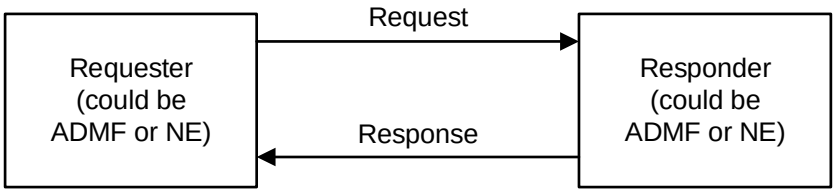

Figure 4.2-1: Showing generic terminology

It is likely that in most situations, the ADMF will initiate the message i.e. to distribute information or request status. However, it is possible that the NE will initiate the request in order to deliver fault reports, etc.

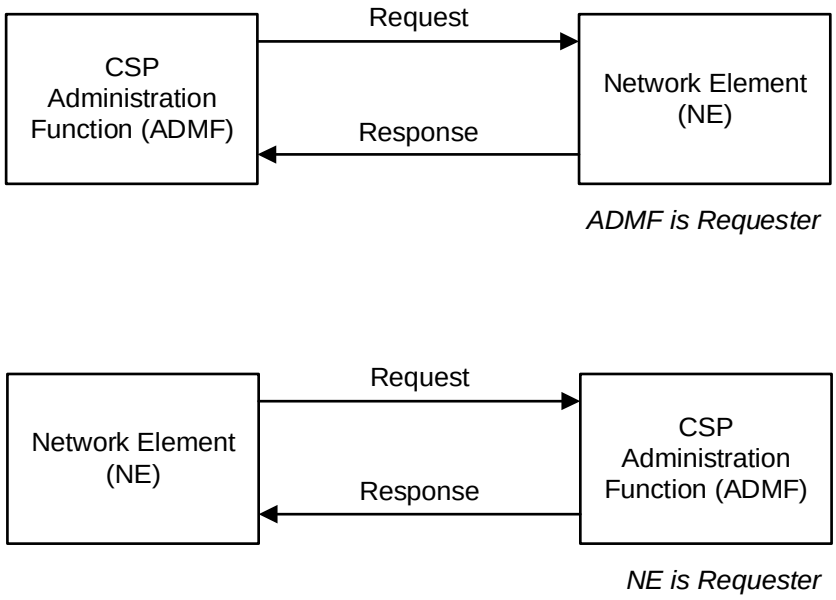

Figure 4.2-2: Showing two situations with either ADMF or NE as the requester

4.3 Overview of security

Security is based on creating public/private keys for the ADMF and each NE for which it is responsible. All transactions over X1 are performed using the security procedures in clause 8, which provide assurance that communication only takes place between an NE and ADMF which have been populated with the relevant key material.

NE implementers are strongly discouraged from exposing additional interfaces for controlling the LI functionality of the NE other than by X1 e.g. via a local administrative interface at the NE. If such additional interfaces exist, any such action performed on the NE shall be captured on the NE audit/logging, and any consequences of such actions shall be able to be seen and controlled by the ADMF that is responsible for the NE i.e. the ADMF shall be able to use the X1 interface to stop or undo any changes made over a local administrative interface. There may be broader consequences that are not covered by the present document if an NE is tasked independently of the X1 interface (e.g. security concerns).

4.4 Relationship to other standards

The present document forms part of a family of internal interface documents covering all of X0, X1, X2 and X3 which are handled as separate standards.

Some models of LI (e.g. 3GPP TS 33.107 [1] and 3GPP TS 33.127 [22]) define interfaces for the purposes described in clause 4.1 (e.g. X1_1, X1_2 and X1_3 defined by 3GPP TS 33.107 [1]; or LI_X1 defined by 3GPP TS 33.127 [22]). The present document is designed to fulfil the requirements for those interfaces.

The present document also specifies the configuration details to be used when ETSI TS 104 000 [30] is used to configure the X1 interface through X0.

4.5 Release management

This clause describes the release management requirements. The requirements are:

- The version of the present document is defined as <major>.<minor>.<patch>.
- The major version should be incremented when making a backwards incompatible change.
- The minor version should be incremented when adding backwards compatible functionality.
- The patch version should be incremented when fixing a backwards compatible bug.

Once a major version has been incremented, the previous major version will be supported for 2 years after publication of the new version. Change requests issued to a version that is no longer supported will need to be issued for the latest supported major version.

5 Basic concepts

5.1 The lifecycle of a Task

5.1.1 Start and end of a Task

A Task relates to a single target identifier, and goes from the point an ActivateTask Request is sent by the ADMF to the time a DeactivateTask Request is sent by the ADMF, a "terminating fault" occurs, or (for Tasks with the "ImplicitDeactivationAllowed" flag set) the NE determines that it has completed.

The present document does not define which situations are categorized as "terminating faults". Local recovery procedures should be followed before a Task is ended with a "terminating fault". In general, irrecoverable failures with an interception, or major security issues at an NE should be considered terminating faults, and certain outcomes with keepalives are also terminating faults (where defined in clause 6.6.2).

5.1.2 Identification of a Task

Each Task on X1 is uniquely identified by an X1 Identifier (XID) and it is handled independently of all others. The ADMF shall assign the XID as a version 4 UUID as per IETF RFC 4122 [2]. The ADMF is responsible for correlating the XID to any LI instance identifiers used to communicate with Law Enforcement. When used between the ADMF and the MDF, the entire LI system may support one of several possibilities:

1) an XID may only map to a single LIID; or
2) an XID may map to multiple LIIDs.

In the first case, each intercept is separately provisioned for a target ID at a given POI. In either case, the ADMF shall provide the XID to LIID(s) mapping to the MDF.

In addition, the XID is released once the Task has ended.

5.1.3 Destinations

Intercepted traffic is delivered by the NE to a Destination. Each Destination is uniquely identified by a Destination Identifier (DID), and is handled independently from details of the Task. DIDs can optionally be grouped with individual DID preference weightings as part of a Destination Set. Destination Sets specify an action, which defines how DIDs within the Destination Set are used; Destination Sets are uniquely identified by their Generic Object ID, which is referred to as the Destination Set Identifier (DSID) (see annex E).

Each Task is associated with one or more Destinations or Destination Sets. Prior to associating a Task with a given DID or DSID, it is required that a Destination with the DID, or Destination Set with the DSID has already been created (as described in clause 6.3 and annex E) but there is no requirement that a connection has been successfully established for that DID or DSID.

Checks regarding availability and status of downstream delivery of information are outside the scope of the present document.

5.1.4 Generic Objects

The NE may require supplementary information which is not described within the Task or Destination objects. Such information is contained within structures derived from a Generic Object, which may be managed via the messages defined in clause 6.8. Generic Object are defined in clause 6.8.1.2. This mechanism shall only be used for the implementation of objects that are defined and standardized in the present document.

5.2 The lifecycle of an X1 request/response

5.2.1 Identification of X1 request/response

Each request and response shall be identified by an X1TransactionID. The requester (may be ADMF or NE) shall assign an X1TransactionID as a version 4 UUID as per IETF RFC 4122 [2].

5.2.2 Responding to the request

The response shall be sent without undue delay and shall be sent within TIME1 of receiving the request. TIME1 shall be configurable and by default TIME1 shall be five seconds. TIME2, the time a requester waits for a response, shall be configurable, it shall be at least twice TIME1 and by default shall be fifteen seconds.

An error response shall be sent if the request is not compliant syntactically (it does not match the schema) or semantically (it is not compliant or consistent with the existing state of the NE e.g. activating an existing XID).

If the request is compliant, one of the following responses shall be sent:

- "OK - Acknowledged and Completed" response shall be sent if the request is fully understood, compliant and the request has been successfully completed. If the request was a request for information then all the information shall be delivered together as part of the "OK - Acknowledged and Completed" response. The NE and ADMF shall be designed so that information requested (status and Task information) is in a data store which is readily available without undue delay and within TIME1.

- If the action requested cannot be completed within TIME1, an "OK - Acknowledged" response shall be sent. A status report shall be sent by the NE as soon as the action is completed or if it is unsuccessful (see clause 6.5.2.2). This status report shall be sent as a new request/response pair, using the same XID or DID but the status report shall have its own X1TransactionID. The "OK - Acknowledged" response shall only be used for responding to requests which are Activating, Modifying or Deleting either Tasks or Destinations (those in clauses 6.2 and 6.3) and they shall not be used to respond to other request types.

5.2.3 Behaviour if a response is not received

If the requester has not received a response after TIME2 (as defined in clause 5.2.2), or if a status report on the completion of the whole request following an "OK - Acknowledge" has not been received in a timely fashion, the requester may assume that either the request or response failed to get through. For example, the requester may consider requesting the status of the XID in question to see whether the prior request has been actioned (e.g. ActivateTask, ModifyTask, DeactivateTask or DeactivateAllTasks) or the requester may re-send the original request (as a new request, with a new X1TransactionID).

5.3 Warnings and Faults

The present document uses the term "error" to mean a protocol error within the X1 protocol as defined in clause 6.7.

All other problems are categorized as warnings, alerts or faults:

- Warnings are one-off problems i.e. sent by the NE and then not referred to again over X1. Warnings shall not be used for issues which are affecting traffic (i.e. losing content or intercept-related information). For example, warnings may include resources being nearly exhausted but not yet traffic-affecting. Warnings should include that keys/certificates are about to expire.
- Alerts are one-off problems that might affect traffic (e.g. cleared database).
- Faults are problems which the NE will continue to be aware of and which the NE is trying to manage and/or rectify. Any issue which loses traffic is categorized as a fault.

Warnings and alerts are reported using issue-reporting messages (clause 6.5) but then are not included in any future Status-Getting messages (see clause 6.4). The NE shall log any warnings and alerts for audit reasons.

The NE shall remember which of the XIDs are in fault and whether the NE itself is in a fault situation. An issue report (see clause 6.5) is required at the start of the fault. The NE shall report faults when responding to the Status-Getting message defined in clause 6.4. The NE shall also indicate that a fault has been cleared (see clauses 6.5.2 and 6.5.3) unless otherwise configured.

6 Message Structure and Data Definitions

6.1 X1 Message details

X1 messages contain information as defined in table 6.1-1 (the information is Mandatory, Optional or Conditional as shown in the last column).

Table 6.1-1: Message details

Field	Description	Format	Mandatory (M), Optional (O) or Conditional (C)
ADMF Identifier	Identifies the ADMF uniquely to the NE. Required to match the details provided by the ADMF's X.509 certificate (see clause 8).	Token as per W3C® Recommendation [3], section 3.4.2. Definition and assignment of identifiers is a deployment issue.	M
NE Identifier	Uniquely identifies the NE to the ADMF. Required to match the details provided by the NE's X.509 certificate (see clause 8).	Token as per W3C® Recommendation [3], section 3.4.2. Definition and assignment of identifiers is a deployment issue.	M
MessageTimestamp	Timestamp indicating the time the message was sent.	See ETSI TS 103 280 [4] Qualified Microsecond Date Time.	M
Version	Version of the present document used for encoding the message.	See clause 4.5.	M
X1TransactionID	Used to correlate Request and Response. Shall be omitted for "TopLevelError" situations as defined below this table but otherwise is mandatory.	An ID as defined in clause 5.2.	C

In addition to the information in table 6.1-1, the X1 Request shall indicate the type of request being made (see RequestMessageType in table 6.7-1 of clause 6.7 for the set of request message types and clauses 6.2 to 6.6 and clause 6.8 for the corresponding details), and contain the appropriate request parameters for that type of request.

If the X1 Request could not be parsed, then the response shall be constructed with an ADMF and NE Identifier (extracting the identifier of the Requester from the X.509 certificate if necessary), MessageTimestamp and Version, and a "TopLevelError" flag but no other information.

If the request could be parsed then the response shall indicate the type of response being returned (see clauses 6.2 to 6.6) and contain the appropriate response parameters for that type of response.

A "RequestContainer" is used to contain one or more requests. All requests in a container are delivered at the same time, from the same Requester and to the same Responder. There is no implication about which order they are processed; for this reason, the ADMF should avoid sending ActivateTask and ModifyTask messages for the same XID in the same RequestContainer. A "ResponseContainer" is used to contain all the responses to the requests in the container. The ordering of these responses does not have a meaning. All responses are sent at the same time, from the same Responder and to the same Requester. The RequestContainer and ResponseContainer shall be used even if there is one request and one response.

For each "OK - Acknowledged" response received for the requests transported by a "RequestContainer", the requester should implement logic to assure the related status report is received and the transaction is completed or initiate a recovery procedure.

6.2 Message definitions: starting, modifying and stopping tasks

6.2.1 ActivateTask

6.2.1.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to add a new Task to an NE.

Table 6.2.1.1-1: ActivateTaskRequest

Field	Description	Format	M/C/O
TaskDetails	Target and interception details	See clause 6.2.1.2.	M

Table 6.2.1.1-2: ActivateTaskResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. Also, it is an error if the XID is already present at the NE.	See clause 6.7.	M

6.2.1.2 TaskDetails

The TaskDetails structure shall include the following.

Table 6.2.1.2-1: TaskDetails

Field	Description	Format	M/C/O
XID	Uniquely identifies the Task. There may be more than one different Task relating to the same target identifier (two distinct XIDs). The X1 interface supports delivery for this situation (i.e. it is not considered an error on the X1 interface).	UUIDv4 (see clause 5.1).	M
TargetIdentifiers	List of criteria which are used to identify the traffic to be intercepted. Where multiple criteria are present, all criteria are required to be matched. If an NE cannot target based on the criteria specified (e.g. due to an unsupported format or inappropriate combination of identifiers) the NE shall reject the request with an appropriate error. It is an implementation decision which identifiers and combinations of identifiers are supported.	Each TargetIdentifier given follows one of the formats given in table 6.2.1.2-2.	M
DeliveryType	Statement of whether to deliver X2 and/or X3. An MDF shall ignore the contents of the field, and use the DeliveryType value given in the relevant MediationDetails structure (see annex C).	Enumerated value - one of "X2Only", "X3Only" and "X2andX3".	M
ListOfDIDs	Details of where to send the intercepted traffic. It is an implementation decision for the NE to determine how to duplicate traffic if multiple destinations and/or destination sets are specified, or if multiple destinations or destination sets are supported.	List of Destination Identifiers (DID) and/or List of Destination Set Identifiers (DSID) referencing the desired delivery destination records.	M
ListOfMediationDetails	Set of details for use by an NE that is performing mediation (i.e. a mediation and delivery function). This shall be included between the ADMF and the MDF. Multiple instances of this parameter may be included (e.g. when multiple LIIDs are associated with an XID).	See annex C.	C
CorrelationID	Correlation identifier to assign to intercepted material for this Task. Intended for use in triggering scenarios, and shall be ignored by non-mediation function NEs.	Unsigned integer.	O

Field	Description	Format	M/C/O
ImplicitDeactivationAllowed	Indication that a Task may implicitly deactivate itself once the NE has determined that it has completed. On deactivation of the Task, the NE shall issue a ReportTaskIssue message with the appropriate TaskReportType (see clause 6.5.2).	Boolean.	O
ProductID	When provided, shall be used by the receiving entity to populate the X2/X3 XID header as per ETSI TS 103 221-2 [19], clause 5.2.7 instead of the XID of the Task. If not provided, the XID of the Task shall be used.	UUIDv4.	O
ListOfServiceTypes	Shall be included when explicitly identifying the CSP-provided service(s) to be reported for this task. Details of the use of this field are left to the relevant LI architecture.	One or more of the enumerated values of the ServiceType field as listed in table C.2.2-2.	C
TaskDetailsExtensions	One or more extension placeholders; each may be populated by a list of elements defined by external specifications.	See annex B.	O
ListOfTrafficPolicyReferences	Ordered list of TrafficPolicyReferences to be applied to the LITaskObject.	See ETSI TS 103 120 [28], clause 8.2.13 ListOfTrafficPolicyReferences.	O

If a Task has an invalid combination of DeliveryType and Destinations (e.g. "X2andX3" delivery specified, but only an X2 Destination given), or an invalid combination of DeliveryType and any Destinations included in the Destination Set identified by the DSID, then the NE shall reject the ActivateTaskRequest with an appropriate error.

If a Task has a ServiceType not supported by the NE, then the NE shall reject the ActivateTaskRequest with an appropriate error. If the expected services to which interception applies are the only services that an NE provides, then inclusion of ServiceType to the LI function in that NE is not necessary. If the ServiceType is not included, then interception applies to all services supported by the NE.

The list of permissible TargetIdentifier formats is given in table 6.2.1.2-2.

Table 6.2.1.2-2: TargetIdentifier Formats

Format Name	Description	Format
E164Number	E.164 Number in fully international format, written as decimal digits.	See ETSI TS 103 280 [4], clause 6.6 InternationalE164.
IMSI	International Mobile Subscriber Identity, following the Recommendation ITU-T E.212 [5] numbering scheme, written as decimal digits.	See ETSI TS 103 280 [4], clause 6.7 IMSI.
IMEI	International Mobile station Equipment Identity, following the numbering plan defined in 3GPP TS 23.003 [6], written as decimal digits without the (Luhn) check digit.	See ETSI TS 103 280 [4], clause 6.8 IMEI.
MACAddress	A MAC address.	See ETSI TS 103 280 [4], clause 6.25 MACAddress.
IPv4Address	An IPv4 address.	See ETSI TS 103 280 [4], clause 6.11 IPv4Address.
IPv6Address	IPv6 address.	See ETSI TS 103 280 [4], clause 6.13 IPv6Address.
IPv4CIDR	IPv4CIDR, written in dotted decimal notation followed by CIDR notation.	See ETSI TS 103 280 [4], clause 6.12 IPv4CIDR.
IPv6CIDR	IPv6CIDR written as eight groups of four hexadecimal digits separated by a colon, followed by CIDR notation.	See ETSI TS 103 280 [4], clause 6.14 IPv6CIDR.
TCPPort	TCP Port number, written in decimal notation.	See ETSI TS 103 280 [4], clause 6.17 TCPPort.
TCPPortRange	Range of TCP Ports, written as decimal numbers separated by a colon.	See ETSI TS 103 280 [4], clause 6.18 TCPPortRange.

Format Name	Description	Format
TCPPortList	List of TCP Ports. A POI shall treat all values within the list with an OR operation for the purpose of target identification.	A list of TCPPort values. See ETSI TS 103 280 [4], clause 6.17 TCPPort.
UDPPort	UDP Port number, written in decimal notation.	See ETSI TS 103 280 [4], clause 6.19 UDPPort.
UDPPortRange	Range of UDP Ports, written as decimal numbers separated by a colon.	See ETSI TS 103 280 [4], clause 6.20 UDPPortRange.
UDPPortList	List of UDP Ports. A POI shall treat all values within the list with an OR operation for the purpose of target identification.	A list of UDPPort values. See ETSI TS 103 280 [4], clause 6.19 UDPPort.
EmailAddress	Email address.	See ETSI TS 103 280 [4], clause 6.26 EmailAddress.
InternationalizedEmailAddress	Email address following IETF RFC 6530 [23].	See ETSI TS 103 280 [4], clause 6.49 InternationalizedEmailAddress.
SIP-URI	SIP-URI according to the SIP URI scheme. See IETF RFC 3261 [7].	See ETSI TS 103 280 [4], clause 6.31 SIPURI.
TEL-URI	TEL-URI according to the TEL URI scheme (see IETF RFC 3966 [8]). Implementers should consider whether the value could be sent as an E.164 number (or one of the related types) instead.	See ETSI TS 103 280 [4], clause 6.32 TELURI.
H323-URI	H323 URI according to the H323 URI scheme (see IETF RFC 3508 [9]).	See ETSI TS 103 280 [4], clause 6.62 H323URI.
IMPU	IP Multimedia Public Identity, as per 3GPP TS 23.003 [6].	See ETSI TS 103 280 [4], clause 6.63 IMPU.
IMPI	IP Multimedia Private Identity, as per 3GPP TS 23.003 [6].	See ETSI TS 103 280 [4], clause 6.64 IMPI.
NAI	Network Access Identifier following IETF RFC 7542 [10] format.	See ETSI TS 103 280 [4], clause 6.47 NAI.
RADIUS	Any Radius attribute that uniquely identifies the subscriber within the specific CSP (see note 1).	Given as binary octets containing RADIUS AVP following IETF RFC 2865 [11], section 5 (see note 2).
GTPUTunnelId	GTP-U Tunnel Identifier.	Given as a 32-bit integer.
GTPCTunnelId	GTP-C Tunnel Identifier.	Given as a 32-bit integer.
CallPartyRole	Identifies the role of a party in a call. Intended for use in conjunction with e.g. E164Number.	One of the values "Originating", "Terminating", "ForwardedTo".
NonLocalIdentifier	Identifies whether the identifier is local or non-local. Intended for use in conjunction with e.g. E164Number.	One of the values "Local" or "NonLocal".
SUPIIMSI	Subscription Permanent Identifier.	See ETSI TS 103 280 [4], clause 6.39 SUPIIMSI.
SUPINAI	Subscription Permanent Identifier.	See ETSI TS 103 280 [4], clause 6.40 SUPINAI.
SUCI	Subscription Concealed identifier.	See ETSI TS 103 280 [4], clause 6.41 SUCI.
PEIIMEI	Permanent Equipment Identifier.	See ETSI TS 103 280 [4], clause 6.42 PEIIMEI.
PEIIMEICheckDigit	Permanent Equipment Identifier.	See ETSI TS 103 280 [4], clause 6.43 PEIIMEICheckDigit.
PEIIMEISV	Permanent Equipment Identifier.	See ETSI TS 103 280 [4], clause 6.44 PEIIMEISV.
GPSIMSISDN	General Purpose Subscription Identifier.	See ETSI TS 103 280 [4], clause 6.45 GPSIMSISDN.
GPSINAI	General Purpose Subscription Identifier.	See ETSI TS 103 280 [4], clause 6.46 GPSINAI.
EUI64	64 bit Extended Unique Identifier following IETF RFC 7042 [27] format.	See ETSI TS 103 280 [4], clause 6.50 EUI64.
ServiceAccessIdentifier	Identifies a user within the context of a service.	See ETSI TS 103 280 [4], clause 6.58 ServiceAccessIdentifier.
HashedIdentifier	Hashed target identifier.	See annex D.

Format Name	Description	Format
TargetIdentifierExtension	Identifier defined by an external specification.	See annex B.
VRF	Identifies a VRF instance within the context of a network.	See ETSI TS 103 280 [4], clause 6.65 VRF.
NOTE 1: Future versions of the present document may need to consider temporary identifiers including pseudonyms or short-term identifiers which have been derived from the permanent identifiers. NOTE 2: Depending on NE implementation, this may not be exactly the same binary representation used to match traffic e.g. for case-insensitive matching.		

6.2.2 ModifyTask

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to modify an existing Task on the NE. All details for the Task shall be given (i.e. the modified details and the information that is unchanged) to totally replace the previous Task details.

Depending on the NE implementation, it may not be possible to modify some or all of the Task details. If the NE cannot modify one or more of the elements in the ModifyTaskRequest, it shall reject the entire ModifyTaskRequest with an appropriate error response.

The length of time an NE requires to make the changes requested in the ModifyTaskRequest message is an implementation detail, but the expectation is that changes are made without undue delay.

Table 6.2.2-1: ModifyTaskRequest

Field	Description	Format	M/C/O
Task details	Target and interception details (same as for ActivateTaskRequest).	See clause 6.2.1.2.	M

Table 6.2.2-2: ModifyTaskResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. Also, it is an error if the XID is not already present.	See clause 6.7.	M

6.2.3 DeactivateTask

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to deactivate (permanently stop and remove) a Task on the NE.

There is no concept of suspension or temporary deactivation. To stop a Task "temporarily", ADMFs shall deactivate the Task and then activate a new Task.

Table 6.2.3-1: DeactivateTaskRequest

Field	Description	Format	M/C/O
XID	See clause 5.1.	See clause 5.1.	M

Table 6.2.3-2: DeactivateTaskResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. Also, it is an error if the XID is not already present at the NE.	See clause 6.7.	M

6.2.4 DeactivateAllTasks

DIRECTION: ADMF to NE.

USAGE: If enabled, the DeactivateAllTasks command shall perform a "DeactivateTask" command for all Tasks on the NE.

Table 6.2.4-1: DeactivateAllTasksRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.2.4-2: DeactivateAllTasksResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. See below regarding whether "DeactivateAllTasks" is enabled; if Disabled then DeactivateAllTasks always triggers an error response of type "DeactivateAllTasks message is not enabled".	See clause 6.7.	M

The DeactivateAllTasks request shall be supported by all implementations of the present document. It should be agreed in advance as to whether the DeactivateAllTasks request is enabled or disabled. By default (if there has been no agreement in advance) then DeactivateAllTasks is enabled.

6.3 Message definitions: creating, modifying and removing Destinations

6.3.1 CreateDestination

6.3.1.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to add a new Destination to the NE.

Table 6.3.1.1-1: CreateDestinationRequest

Field	Description	Format	M/C/O
Destination details	Details of the new destination.	See clause 6.3.1.2.	M

Table 6.3.1.1-2: CreateDestinationResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. Also, it is an error if the DID is already present at the NE.	See clause 6.7.	M

6.3.1.2 DestinationDetails

DestinationDetails relate to the delivery of information from the NE to a Destination.

The DestinationDetails structure is defined as follows.

Table 6.3.1.2-1: DestinationDetails

Field	Description	Format	M/C/O
DID	Destination Identifier which uniquely identifies the destination.	UUIDv4 (see clause 5.1).	M
FriendlyName	A human-readable name associated with the delivery destination.	Free-text string.	O
DeliveryType	Statement of whether to deliver X2 and/or X3 to this destination.	Enumerated value - one of "X2Only", "X3Only" and "X2andX3".	M
DeliveryAddress	One of the values from table 6.3.1.2-2 shall be included.	As defined in table 6.3.1.2-2.	M
DestinationDetails Extensions	One or more extension placeholders; each may be populated by a list of elements defined by external specifications.	See annex B.	O

The DeliveryAddress structure is defined as follows.

Table 6.3.1.2-2: DeliveryAddress

Field	Description	Format
IPAddressAndPort	This covers both IPv4 and IPv6 and contains a single IP Address and Port.	IPAddressAndPort from ETSI TS 103 280 [4].
E164Number	E.164 destination.	InternationalE164 (see ETSI TS 103 280 [4]).
URI	URI destination (e.g. an FQDN or other form of URI).	anyURI (see W3C® Recommendation [3], section 3.2.17).
EmailAddress	Email address of the destination.	EmailAddress (see ETSI TS 103 280 [4]).

6.3.2 ModifyDestination

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to modify an existing Destination on the NE. All details for the Destination shall be given (i.e. the modified details and the information that is unchanged) to totally replace the previous Destination details.

Depending on the NE implementation, it may not be possible to modify some or all Destination details while the Destination is in use. If the NE cannot modify one or more of the elements in the ModifyDestinationRequest, it shall reject the entire ModifyDestinationRequest with an appropriate error response.

The length of time an NE requires to make the changes requested in the ModifyDestinationRequest message is an implementation detail, but the expectation is that changes are made without undue delay.

Table 6.3.2-1: ModifyDestinationRequest

Field	Description	Format	M/C/O
DestinationDetails	Updated details for the destination.	See clause 6.3.1.2.	M

Table 6.3.2-2: ModifyDestinationResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. Also, it is an error if the DID is not present.	See clause 6.7.	M

6.3.3 RemoveDestination

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to remove a Destination from the NE.

A Destination may only be removed if it is not referenced by any Tasks or Destination Sets. An NE shall respond with an appropriate error if the ADMF attempts to remove a Destination that is referenced by a Task or Destination Set.

Table 6.3.3-1: RemoveDestinationRequest

Field	Description	Format	M/C/O
DID	See clause 5.1.	See clause 5.1.	M

Table 6.3.3-2: RemoveDestinationResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. Also, it is an error if the DID is not already present at the NE	See clause 6.7	M

6.3.4 RemoveAllDestinations

DIRECTION: ADMF to NE.

USAGE: To completely and permanently remove all Destinations on the NE.

Table 6.3.4-1: RemoveAllDestinationsRequest

Field	Description	Format	M/C/O
There shall be no message parameters.			

Table 6.3.4-2: RemoveAllDestinationsResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. See below regarding whether "RemoveAllDestinations" is enabled; if Disabled then RemoveAllDestinations always triggers an error response.	See clause 6.7.	M

The RemoveAllDestinations request shall be supported by all implementations of the present document.

It shall be agreed in advance as to whether the RemoveAllDestinations request is enabled or disabled. By default (if there has been no agreement in advance) then RemoveAllDestinations is enabled.

If RemoveAllDestinations is disabled, then a RemoveAllDestinations request shall always trigger an ErrorResponse indicating "RemoveAllDestinations request is not enabled".

If RemoveAllDestinations is enabled, then a RemoveAllDestinations request shall remove all Destinations on that NE, or it shall trigger an error for the general error conditions listed in clause 6.7. Since a RemoveDestination request can only be issued against destinations that are not in use, an NE shall respond with an error if the ADMF sends a RemoveAllDestinations request while any of the Destinations are referenced by Tasks or Destination Sets.

6.4 Message details: getting information from NE

6.4.1 Overview

This clause defines messages for the ADMF to request status information from the NE. This is distinct from "Reporting Issues" where the NE pushes information to the ADMF (see clause 6.5).

The following requests and responses shall be supported:

- GetTaskDetails: to request details of a single Task.
- GetDestinationDetails: to request details of a single Destination.
- GetNEStatus: to request status of the NE itself.
- GetAllDetails: requests details of all Tasks, Destinations, Generic Objects and the status of the NE itself.
- GetAllTaskDetails: requests details of all Tasks.
- GetAllDestinationDetails: requests details of all Destinations.
- GetAllGenericObjectDetails: requests details of all Generic Objects.
- ListAllDetails: requests the XIDs of all Tasks, DIDs of all Destinations and Object IDs of all Generic Objects (i.e. not all the details).

6.4.2 GetTaskDetails

6.4.2.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to retrieve the details of a particular Task.

Table 6.4.2.1-1: GetTaskDetailsRequest

Field	Description	Format	M/C/O
XID	See clause 5.1.	See clause 5.1.	M

Table 6.4.2.1-2: GetTaskDetailsResponse

Field	Description	Format	M/C/O
TaskResponseDetails	The Task details are as per clause 6.2.1.2, additionally containing a TaskStatus structure as per clause 6.4.2.2, unless there is an error, in which case see clause 6.7. If the XID is not present, this is an error (the appropriate error code shall be used, see clause 6.7).	See clauses 6.2.1.2 and 6.4.2.2.	M

6.4.2.2 TaskStatus

The TaskStatus contains information about a Task as collected internally by the NE.

Table 6.4.2.2-1: TaskStatus

Field	Description	Format	M/C/O
ProvisioningStatus	Indicates whether the Task has been provisioned ("complete"), has failed to provision ("failed") or whether it is awaiting provisioning ("awaitingProvisioning").	One of the values "awaitingProvisioning", "failed" or "complete".	M

Field	Description	Format	M/C/O
ListOfFaults	List of all active faults on that Task. If there are no faults, the listOfFaults field shall be encoded without containing any unresolvedFault tags.	List of ErrorInformation structures (see clause 6.7).	M
TimeOfLastIntercept	Time of last traffic intercepted if any (omit if none seen so far or as provided beneath this table). This time may also be updated periodically (instead of per packet) if required due to performance reasons.	See ETSI TS 103 280 [4], Qualified Microsecond Date Time.	C
AmountOfX2Data	Data transmitted over X2 since the creation of the Task in bytes, summed across all Destinations. This field shall be included unless the exception beneath this table applies. If given, shall be correct at the time given in TimeOfLastIntercept.	Integer.	C
AmountOfX3Data	Data transmitted over X3 since the creation of the Task in bytes, summed across all Destinations. This field shall be included unless the exception beneath this table applies. If given, shall be correct at the time given in TimeOfLastIntercept.	Integer.	C
TimeOfLastModification	Time of the last modification to the Task (omit only if unmodified or as provided beneath this table).	See ETSI TS 103 280 [4], Qualified Microsecond Date Time.	C
NumberOfModifications	Number of successful modifications since start. This field shall be included unless the exception beneath this table applies.	Integer.	C
TaskStatusExtensions	One or more extension placeholders; each may be populated by a list of elements defined in external specifications.	See annex B.	O

For any of the following fields: TimeOfLastIntercept, AmountOfX2Data, AmountOfX3Data, TimeOfLastModification and NumberOfModifications, if the functionality needed to determine information for a field is not implemented by an NE, the field shall always be omitted.

6.4.3 GetDestinationDetails

6.4.3.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to retrieve the details of a particular Destination.

Table 6.4.3.1-1: GetDestinationRequest

Field	Description	Format	M/C/O
DID	See clause 5.1.	See clause 5.1.	M

Table 6.4.3.1-2: GetDestinationResponse

Field	Description	Format	M/C/O
DestinationResponseDetails	The destination details are as per table 6.3.1.2-1, additionally containing a DestinationStatus structure as per clause 6.4.3.2, unless there is an error, in which case see clause 6.7. If the DID is not present, this is an error (the appropriate error code shall be used, see clause 6.7).	See clauses 6.3.1.2 and 6.4.3.2.	M

6.4.3.2 DestinationStatus

The DestinationStatus relates only to the status of the delivery Destination as seen by the NE.

Table 6.4.3.2-1: DestinationStatus

Field	Description	Format	M/C/O
DestinationStatus	Status of Destination. Indicating whether the destination is active and working, or whether there is a delivery fault and traffic being lost. It is possible in the DeliveryFault state that some traffic is still being delivered - the determining factor is that issues with delivery to this destination is causing some traffic to be lost.	One of "ActiveAndWorking" or "DeliveryFaults".	M
ListOfFaults	List of all active faults on that Destination.	List of ErrorInformation structures (see clause 6.7).	M

6.4.4 GetNEStatus

6.4.4.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to determine the status of the NE.

Table 6.4.4.1-1: GetNEStatusRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.4.4.1-2: GetNEStatusResponse

Field	Description	Format	M/C/O
NEStatusDetails	The NEStatusDetails for the NE. The NE Status shall be one of "OK" i.e. no NE faults, or "Faults" i.e. NE losing traffic (these are separate from delivery faults which are reported per XID). Additionally, a list of currently unresolved faults (list of ErrorInformation items) shall be included (previous warnings are not included here).	Enumerated NEStatus value - one of "OK" or "Faults". List of ErrorInformation structures (see clause 6.7).	M

6.4.5 GetAllDetails

6.4.5.1 Summary

DIRECTION: The GetAllDetails command goes from ADMF to NE.

USAGE: For the ADMF to determine the details of all Tasks, Destinations and the status of the NE itself.

Table 6.4.5.1-1: GetAllDetailsRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.4.5.1-2: GetAllDetailsResponse

Field	Description	Format	M/C/O
NEStatusDetails	The NEStatusDetails for the NE. The NEStatus shall be one of "OK" i.e. no NE faults, or "Faults" i.e. NE losing traffic (these are separate from delivery faults which are reported per XID). Additionally, a list of currently unresolved faults (list of ErrorInformation items) shall be included (previous warnings are not included here).	Enumerated NEStatus value - one of "OK" or "Faults". List of ErrorInformation structures (see clause 6.7).	M
ListOfTaskResponseDetails	The response shall include TaskResponseDetails structures for all Tasks present on the NE. If there are no Tasks, an empty list shall be returned - this is not an error.	See clauses 6.2.1.2 and 6.4.2.2.	M
ListOfDestinationResponse Details	The response shall include DestinationResponseDetails structures for all destinations present on the NE. If there are no destinations, an empty list shall be returned - this is not an error.	See clauses 6.3.1.2 and 6.4.3.2.	M
ListOfGenericObjectDetails	The response shall include Generic Object details for every object present on the NE. If there are no such objects, an empty list shall be returned - this is not an error. May be omitted if Generic Objects are not supported by the NE.	See clause 6.8.	C

6.4.6 ListAllDetails

6.4.6.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to retrieve the list of all XIDs and DIDs (i.e. a list of identifiers) but no details.

Table 6.4.6.1-1: ListAllDetailsRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.4.6.1-2: ListAllDetailsResponse

Field	Description	Format	M/C/O
ListOfXIDs	A list of all XIDs on the NE. If there are none, then an empty list is returned; this is not an error.	List of XIDs.	M
ListOfDIDs	A list of all DIDs on the NE. If there are none, then an empty list is returned; this is not an error.	List of DIDs.	M
ListOfGenericObjectIDs	A list of all Generic Object IDs on the NE (see clause 6.8.1.3). If there are none, an empty list is returned - this is not an error. May be omitted if Generic Objects are not supported by the NE.	List of objectIDs.	C

6.4.7 GetAllTaskDetails

6.4.7.1 Summary

DIRECTION: The GetAllTaskDetails command goes from ADMF to NE.

USAGE: For the ADMF to determine the details of all Tasks.

Table 6.4.7.1-1: GetAllTaskDetailsRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.4.7.1-2: GetAllTaskDetailsResponse

Field	Description	Format	M/C/O
ListOfTaskResponseDetails	The response shall include TaskResponseDetails structures for all Tasks present on the NE. If there are no Tasks, an empty list shall be returned - this is not an error.	See clauses 6.2.1.2 and 6.4.2.2.	M

6.4.8 GetAllDestinationDetails

6.4.8.1 Summary

DIRECTION: The GetAllDestinationDetails command goes from ADMF to NE.

USAGE: For the ADMF to determine the details of all Destinations.

Table 6.4.8.1-1: GetAllDestinationDetailsRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.4.8.1-2: GetAllDestinationDetailsResponse

Field	Description	Format	M/C/O
ListOfDestinationResponse Details	The response shall include DestinationResponseDetails structures for all destinations present on the NE. If there are no destinations, an empty list shall be returned - this is not an error.	See clauses 6.3.1.2 and 6.4.3.2.	M

6.4.9 GetAllGenericObjectDetails

6.4.9.1 Summary

DIRECTION: The GetAllGenericObjectDetails command goes from ADMF to NE.

USAGE: For the ADMF to determine the details of all Generic Objects.

Table 6.4.9.1-1: GetAllGenericObjectDetailsRequest

Field	Description	Format	M/C/O
ObjectType	If present, only the specific XSD type is required rather than the whole Generic Object Details.	URIQualifiedName (as defined in XPath 3.1 [24], definition 117.	O

Table 6.4.9.1-2: GetAllGenericObjectDetailsResponse

Field	Description	Format	M/C/O
ListOfGenericObjectDetails	The response shall include Object details for every object present on the NE or details for object of the specified objectType (if present in the request). If there are no such objects, an empty list shall be returned - this is not an error. May be omitted if Generic Objects are not supported by the NE.	See clause 6.8.	C

6.5 Message details: reporting issues from the NE

6.5.1 Overview

This clause defines request types for the NE to report issues to the ADMF. It is distinct from "Getting Status", in which the ADMF retrieves information from the NE (see clause 6.4).

Issues may be:

- Relating to a particular XID (including delivery issues with that XID).
- Relating to a particular DID.
- Relating to the whole NE.

6.5.2 ReportTaskIssue on given XID

6.5.2.1 Summary

DIRECTION: NE to ADMF.

USAGE: The NE shall send a ReportTaskIssue request when it becomes aware of an issue (warning or fault) relating specifically to a particular XID. It shall also be used to follow up on an "OK - Acknowledged" response, to signal that a request has been completed (clause 5.2) successfully or unsuccessfully.

Faults and warnings are defined in clause 5.3; see also clause 5.1 about terminating and non-terminating faults.

If a non-terminating fault becomes terminating, the NE shall send another ReportTaskIssue.

If a non-terminating fault is cleared, the NE shall send another ReportTaskIssue indicating the fault is cleared.

Table 6.5.2.1-1: ReportTaskIssueRequest

Field	Description	Format	M/C/O
XID	See clause 5.1.	See clause 5.1.	M
TaskReportType	Type of Issue.	See clause 6.5.2.2.	M
TaskIssueErrorCode	Error code associated with the issue, if appropriate.	See clause 6.7.	O
TaskIssueDetails	Further description of issue if appropriate.	Free text.	O
TaskIssueExtensions	One or more extension placeholders; each may be populated by a list of elements defined in external specifications.	See annex B.	O

Table 6.5.2.1-2: ReportTaskIssueResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply.	See clause 6.7.	M

It is possible that the ADMF is not aware of the XID which is referenced in the NE message. The ADMF shall not send an error back to the NE in this situation: it is for the ADMF to decide how to handle this (e.g. GetAllDetails or GetAllTaskDetails or Deactivate the XID in question are possible approaches).

6.5.2.2 Task report types

The TaskReportType shall be one of the following:

- All clear: non-terminating fault resolved.
- Warning: not traffic-affecting.
- Non-terminating fault (e.g. currently unable to collect traffic but not terminating).
- Terminating fault. The message is used by the NE to indicate that the Task has experienced a terminating fault and has been deactivated.
- Implicit Deactivation: A Task with the "ImplicitDeactivationAllowed" flag has been deactivated.
- Actioned: Request has been fully actioned and was successful (to follow up on "OK - Acknowledged" response from clause 5.2).
- Failed: Request has been fully actioned but was unsuccessful (to follow up on "OK - Acknowledged" response from clause 5.2). This is a terminating fault.

6.5.3 ReportDestinationIssue on given DID

6.5.3.1 Summary

DIRECTION: NE to ADMF.

USAGE: The NE shall send a ReportDestinationIssue request when it becomes aware of an issue (warning or fault) relating specifically to a particular DID. It shall also be used to follow up on an "OK - Acknowledged" response, to signal that a request has been completed (clause 5.2) successfully or unsuccessfully.

Faults and warnings are defined in clause 5.3; see also clause 5.1 about terminating and non-terminating faults.

If a non-terminating fault becomes terminating, the NE shall send another ReportDestinationIssue.

If a non-terminating fault is cleared, the NE shall send another ReportDestinationIssue indicating the fault is cleared.

Table 6.5.3.1-1: ReportDestinationIssueRequest

Field	Description	Format	M/C/O
DID	See clause 5.1.	See clause 5.1.	M
DestinationReportType	Type of Issue.	Same as TaskReportType, see clause 6.5.2.2.	M
DestinationIssueErrorCode	Error code for the issue, if appropriate.	See clause 6.7.	O
DestinationIssueDetails	Further description of issue if appropriate.	Free text.	O

Table 6.5.3.1-2: ReportDestinationIssueResponse

Field	Description	Format	M/C/O
OK or Error.	The general errors in clause 6.7 apply.	See clause 6.7.	M

6.5.4 ReportNEIssue

DIRECTION: NE to ADMF.

USAGE: The NE shall send a ReportNEIssue request when it becomes aware of an issue (warning, alert or fault) relating to the whole NE.

NE issues can relate to:

- Any hardware or software issue on NE (storage nearly full, power issue).
- Current security issue on NE.
- Any issues with logging or audit material.
- Any report from manual changes to NE configuration.
- Any report of databases being cleared in the NE.

Table 6.5.4-1: ReportNEIssueRequest

Field	Description	Format	M/C/O
TypeOfNEIssueMessage	Indicates the type of message (Warning, Fault Cleared, Fault Report, Alert).	One of the following: "Warning", "FaultCleared", "FaultReport", "Alert".	M
Description	Description of the issue being reported.	Free text.	M
IssueCode	Integer code indicating the distinct issue information if TypeOfNEIssueMessage is: • "Alert" and the error code is part of the issue codes section in table 6.7-3. • "FaultReport", "FaultCleared" or "Warning" and the Error Code is part of the status/fault codes section in table 6.7-3. The use of this field is mandatory when required by any clause of the present document.	Integer.	C
NEIssueExtensions	One or more extension placeholders; each may be populated by a list of elements defined in external specifications.	See annex B.	O

Table 6.5.4-2: ReportNEIssueResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply.	See clause 6.7.	M

6.6 Message details: pings and keepalives

6.6.1 Ping

DIRECTION: Either direction.

USAGE: At any time from the ADMF or NE, to get a response over the X1 interface (does not test X2 or X3 or onward delivery).

Table 6.6.1-1: PingRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.6.1-2: PingResponse

Field	Description	Format	M/C/O
OK or Error	The OK response has no other content. The general errors in clause 6.7 apply.	See clause 6.7.	M

6.6.2 Keepalive

DIRECTION: The Keepalive command goes from ADMF to NE.

USAGE: See below.

Table 6.6.2-1: KeepaliveRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.6.2-2: KeepaliveResponse

Field	Description	Format	M/C/O
OK or Error	The OK message has no other content. The general errors in clause 6.7 apply.	See clause 6.7.	M

The Keepalive functionality shall be supported by NE and ADMF. It is for prior agreement to determine whether Keepalives are enabled or disabled. By default (with no prior agreement) they are enabled. It is intended as a means for the NE application to assert that the ADMF application is still operational, and, unless otherwise configured, remove all tasking information as a security measure if it is not.

If Keepalives are enabled, the ADMF shall send out a Keepalive message at least every TIME_P1 (by default TIME_P1 is 1 minute) if no other X1 request has been sent to the NE.

If Keepalives are enabled, the NE shall respond with an OK for each Keepalive. The NE shall utilize a timer P2, with a value TIME_P2 (by default TIME_P2 is 1 hour), which is used to determine when the last Keepalive was seen. The NE implementation shall reset the timer P2 whenever any X1 Request is received from the ADMF (including a Keepalive Request). If the NE has not seen a Keepalive message for TIME_P2 (i.e. timer P2 expires) then the NE shall:

- Send a ReportNEIssue request to the ADMF indicating "FaultReport", with a status/fault code 9050 "Keepalives not received" and start a timer P1, with a value TIME_P1 (by default TIME_P1 is 1 minute). In addition to the actions specified below, if at any point in time the NE receives a X1 request (including Keepalive request) from the ADMF, unless otherwise configured, the NE shall clear the error by sending a ReportNEIssue request to the ADMF indicating "FaultCleared":

  - If a Keepalive message or any other X1 request is received from the ADMF before the expiry of timer P1, the NE shall stop the timer P1, reset the timer P2 and continue with normal operations, skipping the actions below.

  - If a ReportNEIssue OK response is received from the ADMF before the expiry of timer P1, the NE shall stop timer P1.

    If the NE is configured to allow deactivation of all tasks as part of the Keepalive procedure,the NE shall utilize a timer P3, with a value TIME P3 (by default TIME P3 is 2 hours), which is started by the NE when the ReportNEIssue OK response is received from the ADMF and is used to determine when the deactivation of all tasks shall occur. If a Keepalive message or any other X1 request is received from the ADMF before the timer P3 expires, the NE shall stop the timer P3, reset the timer P2 and continue normal operation. If the timer P3 expires, the NE shall perform a DeactivateAllTasks command, i.e. deactivate all XIDs on the NE, and should send a ReportNEIssue to the ADMF indicating "Alert" with an issue code 10000 "Database cleared".

    If the NE is configured to not allow deactivation of all tasks as part of the Keepalive procedure, the NE shall reset the timer P2 and continue normal operation. Further steps the NE can take are out of scope of the present document.

- If a ReportNEIssue response is not received from the ADMF (i.e. timer P1 expires) or an error response is received from the ADMF as answer to the ReportNEIssue request and no X1 request (including Keepalive request) was received from the ADMF while timer P1 was running, if the NE is configured to allow the deactivation of all tasks as part of the Keepalive procedure, the NE shall perform a DeactivateAllTasks command i.e. deactivate all XIDs on the NE, and should send a ReportNEIssue to the ADMF indicating "Alert" with an issue code 10000 "Database cleared".

  If instead the NE is configured to not allow deactivation of all tasks as part of the Keepalive procedure, the NE shall reset the timer P2 and continue normal operation. Further steps the NE can take are out of scope of the present document.

6.7 Protocol error details

If the Responder is unable to perform an action requested as part of a Request Message, then it shall respond to that Request Message with an Error Response.

An ErrorResponse is a response which has the information from clause 6.1, but the response body has an error code from the list below and a free text field for further information. It has the following structure.

Table 6.7-1: ErrorResponse

Field	Description	Format	M/C/O
RequestMessageType	Indicates the type of Request Message that the Error Response message is a response to.	One of the following: "ActivateTask", "ModifyTask", "DeactivateTask", "DeactivateAllTasks", "GetTaskDetails", "CreateDestination", "ModifyDestination", "RemoveDestination", "RemoveAllDestinations", "GetDestinationDetails", "GetNEStatus", "GetAllDetails", "GetAllTaskDetails", "GetAllDestinationDetails", "GetAllGenericObjectDetails", "ListAllDetails", "ReportTaskIssue", "ReportDestinationIssue", "ReportNEIssue", "Ping", "Keepalive", "ExtendedRequestMessageType".	M
ErrorInformation	Error code and optional description for the error.	ErrorInformation as defined in table 6.7-2.	M
ExtensionInformation	Indicates the specification of the extension and the extended type of the Request Message that the Error Response message is a response to if the RequestMessageType is "ExtendedRequestMessageType".	ExtensionInformation as defined in table 6.7-4.	C

Table 6.7-2: ErrorInformation

Field	Description	Format	M/C/O
ErrorCode	Integer code indicating the type of error (see table 6.7-3).	Integer.	M
ErrorDescription	Free text field giving further details of the error. Implementers are encouraged to avoid placing sensitive information (such as personally identifiably information or sensitive details of the network) in error messages.	UTF-8 string.	M

The ErrorResponse is used only as a response to a request which could not be actioned or understood. It is different from reporting on the status of the Task which are called "faults" and "warnings" but not "protocol errors".

Table 6.7-3: Error codes

Error Code	Error Description	Suggested Information elements
General message errors		
1000	Generic error	Details of the error
1010	Syntax/schema error	Details of the schema or syntax error

Error Code	Error Description	Suggested Information elements
1020	Unsupported version	Version supported by the issuing system
1030	ADMF Identifier does not match certificate details	None
1040	Unexpected ADMF Identifier	None
1050	NE Identifier does not match certificate details	None
1060	Unexpected NE Identifier	None
1070	Keepalive not supported	None
1080	Unsupported request	None
Identifier errors		
2010	XID already exists on NE	XID in question
2020	XID does not exist on NE	XID in question
2030	DID already exists on the NE	DID in question
2040	DID does not exist on the NE	DID in question
2050	GenericObjectID already exists on the NE	GenericObjectID in question
2060	GenericObjectID does not exist on the NE	GenericObjectID in question
ActivateTask/ModifyTask errors		
3000	Generic ActivateTask failure	Details of why the Task cannot be activated
3001	Generic ModifyTask failure	Details of why the Task cannot be modified
3010	Unsupported TargetIdentifier type	Details of the unsupported TargetIdentifier type
3020	Unsupported combination of TargetIdentifiers	Details of the unsupported combination
3030	Multiple destinations not supported	None
3040	Invalid combination of DeliveryType and Destinations specified	None
3050	Unsupported ServiceType	Details of the unsupported ServiceType
DeactivateTask failures		
4000	Generic DeactivateTask failure	Details of why the Task cannot be deactivated
DeactivateAllTasks failures		
5000	Generic DeactiveAllTasks failure	Details of why all Tasks cannot be removed
5010	DeactivateAllTasks not enabled	None
CreateDestination/ModifyDestination failures		
6000	Generic CreateDestination failure	Details of why the Destination cannot be created
6001	Generic ModifyDestination failure	Details of why the Destination cannot be modified
6020	Unsupported DeliveryAddress type	Details of the DeliveryAddress type requested
RemoveDestination failures		
7000	Generic RemoveDestination failure	Details of why the Destination cannot be removed
7010	Destination in use	Details of the Task(s) and/or of the Destination Set referencing the Destination if possible
RemoveAllDestinations failures		
8000	Generic RemoveAllDestinations failure	Details of why all Destinations cannot be removed
8010	Destinations in use	Details of which Destinations are in use, and (if possible) by which Tasks and/or of the Destination Sets
8020	RemoveAllDestinations not enabled	None
Generic Object failures		
8500	Generic CreateObject failure	Details of why the Generic Object cannot be created
8510	Generic ModifyObject failure	Details of why the Generic Object cannot be modified
8530	Generic DeleteObject failure	Details of why the Generic Object cannot be deleted
8540	Generic DeleteAllObjects failure	Details of why all Generic Objects cannot be deleted
8550	DeleteAllObjects not enabled	None
Status/fault codes		
9000	Error cleared	Nature of the error which has now cleared
9010	Generic warning	Details of the warning
9020	Generic non-terminating fault	Details of the fault
9030	Terminating fault	Details of the fault
9040	Request actioned	X1TransactionID of the request now actioned
9050	Keepalives not received	None
Issue codes		
10000	Database cleared	None

Implementers shall use the most specific error code available.

Table 6.7-4: ExtensionInformation

Field	Description	Format	M/C/O
ExtensionSpecification	Value indicating the specification using the extension.	One of the following: "TS133128".	M
ExtendedRequestMessageType	Free text field giving the name of the extended type of Request Message that the Error Response message is a response to, as defined in the related specification.	UTF-8 string.	M

6.8 Message definitions: managing general objects

6.8.1 CreateObject

6.8.1.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to add a new Generic Object to the NE.

Table 6.8.1.1-1: CreateObjectRequest

Field	Description	Format	M/C/O
createObject	Contains a structure derived from the GenericObject.XSD type.	See clause 6.8.1.2.	M

Table 6.8.1.1-2: CreateObjectResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply.	See clause 6.7.	M

Generic Objects provide a means of storing additional information at the NE beyond that described by Tasks or Destinations. If the NE already contains a Generic Object with the same objectID as the one supplied in the CreateObjectRequest, the NE shall reject the request with an appropriate error response. If the NE cannot store the supplied record e.g. because it does not support the supplied object type, it shall reject the CreateObjectRequest with an appropriate error response.

6.8.1.2 Generic Object Structure

All Generic Objects shall be descended from the abstract X1Object defined in the schema of the present document. The X1Object definition contains the following fields.

Table 6.8.1.2-1: X1Object

Field	Description	Format	M/C/O
objectID	Shall uniquely identify the Generic Object at the NE.	GenericObjectID (see clause 6.8.1.3).	M

6.8.1.3 GenericObjectID

A GenericObjectID uniquely identifies a given Generic Object. Derived Generic Object types may introduce further identifier fields, but the GenericObjectID shall be unique for that object at the NE, and shall be the identifier used in relevant Generic Object messages (see clause 6.8).

The GenericObjectID shall be given as a UUID.

6.8.2 ModifyObject

6.8.2.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to modify an existing Generic Object on the NE. All the details for the object shall be given (i.e. the modified details and the information that is unchanged) to totally replace the previous object details.

Table 6.8.2.1-1: ModifyObjectRequest

Field	Description	Format	M/C/O
modifyObject	Contains a structure derived from the GenericObject XSD type.	See clause 6.8.1.2.	M

Table 6.8.2.1-2: ModifyObjectResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply.	See clause 6.7.	M

Depending on the NE implementation, it may not be possible to modify some or all of the object details. If the NE cannot modify one or more of the elements in the modifyObject structure, it shall reject the entire ModifyObjectRequest with an appropriate error response.

6.8.3 DeleteObject

6.8.3.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to remove a Generic Object from the NE.

Table 6.8.3.1-1: DeleteObjectRequest

Field	Description	Format	M/C/O
objectID	Unique identifier for the object.	See clause 6.8.1.3.	M

Table 6.8.3.1-2: DeleteObjectResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply.	See clause 6.7.	M

6.8.4 GetObject

6.8.4.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to retrieve details of a particular Generic Object from the NE.

Table 6.8.4.1-1: GetObjectRequest

Field	Description	Format	M/C/O
objectID	Unique identifier for the object.	See clause 6.8.1.3.	M

Table 6.8.4.1-2: GetObjectResponse

Field	Description	Format	M/C/O
genericObjectResponseDetails	Structure containing the object and its current status.	See table 6.8.4.1-3.	M

Table 6.8.4.1-3: GenericObjectResponseDetails

Field	Description	Format	M/C/O
object	The details of the object identified by the objectID, unless there is an error, in which case see clause 6.7. If no object with the supplied objectID is present, this is an error.	See clause 6.8.1.2.	M
status	Types derived from GenericObject may also derive status definitions from the GenericObjectStatus type. In this case, the relevant derived status shall be supplied here, otherwise the field shall be omitted.	Derived type.	C

6.8.5 ListObjectsOfType

6.8.5.1 Summary

DIRECTION: ADMF to NE.

USAGE: Used by the ADMF to retrieve all identifiers (objectIDs) of objects stored at the NE that have a particular type.

Table 6.8.5.1-1: ListObjectsOfTypeRequest

Field	Description	Format	M/C/O
objectType	Shall be set to the name of the specific XSD type required.	URIQualifiedName (as defined in XPath 3.1 [24], definition 117).	M

Table 6.8.5.1-2: ListObjectsOfTypeResponse

Field	Description	Format	M/C/O
listOfObjects	List of objectIDs corresponding to those objects stored at the NE which are of the type specified in the query.	See clause 6.8.1.3.	M

6.8.6 DeleteAllObjects

6.8.6.1 Summary

DIRECTION: ADMF to NE.

USAGE: If enabled, the DeleteAllObjects command shall perform a "DeleteObject" command for all Generic Objects on the NE.

Table 6.8.6.1-1: DeleteAllObjectsRequest

Field	Description	Format	M/C/O
There shall be no request parameters.			

Table 6.8.6.1-2: DeleteAllObjectsResponse

Field	Description	Format	M/C/O
OK or Error	The general errors in clause 6.7 apply. See below regarding whether "DeleteAllObjects" is enabled; if Disabled then DeleteAllObjects always triggers an error response of type "DeleteAllObjects message is not enabled".	See clause 6.7.	M

The DeleteAllObjects request shall be supported if the implementation supports Generic Objects. It should be agreed in advance as to whether the DeleteAllObjects request is enabled or disabled. By default (if there has been no agreement in advance) then DeleteAllObjects is enabled.

7 Transport and Encoding

7.1 Introduction

The present document defines a single profile for transport and encoding of X1 messages.

7.2 Profile A

7.2.1 Encoding

XML encoding shall be used. An XML schema (XSD) is provided in archive ts_10322101v012301p0.zip which accompanies the present document. In the event of a discrepancy between the XSD and encoding requirements that are stated in the present document, the XSD shall be considered authoritative. With the XSD being authoritative, change requests to the present document shall be accompanied with examples that cover all XSD changes in order to ensure the XML examples validate against the XSD.

Implementers on both the sending and receiving end shall validate the XML they generate against the XSD. As this may incur a performance cost in production environments, validation shall be performed during the engineering/implementation phase and optionally in development, integration or production environments.

Samples, which provide an informative example for implementations of the present document, are available together with the normative XSD at https://forge.etsi.org/rep/li/schemas-definitions/-/tree/spec/103221-1/1.23.1/103221-1?ref_type=tags. The samples do not form part of the normative specification.

7.2.2 Transport layer

7.2.2.1 HTTPS and HTTP

HTTPS shall be used as per IETF RFC 2818 [12]. The details relating to HTTP are given in this clause and the details relating to TLS are specified in clause 8.2.

In this clause, the term HTTP is used (it is implicit that it is in fact HTTPS i.e. that the HTTP is used over TLS).

7.2.2.2 How HTTP is used

The ADMF and NE shall both run HTTP clients and servers:

- For messages where the ADMF is the requester, the ADMF shall use its HTTP client and the NE shall use its HTTP server.
- For messages where the NE is the requester, the NE shall use its HTTP client and the ADMF shall use its HTTP server.

Details in the request:

- Each "RequestContainer" shall be sent as a HTTP request. It shall be a "POST" message (regardless of which type of X1 request it is) and the message body shall contain the RequestContainer as described in clause 6.

Details in the response:

- Each "ResponseContainer" message shall be sent as a HTTP response.
- The response shall indicate HTTP level errors within the range of HTTP error codes. If the HTTP level transaction is successful, then the response shall be a 200 OK message, with the ResponseContainer contained within the message body.
- HTTP error codes shall only be used to indicate HTTP-level errors, and shall not be used to indicate errors with the X1 responses themselves. X1-level errors shall be indicated by correct use of the appropriate X1 ErrorResponse, encoded and returned as a HTTP 200 OK response.

7.2.2.3 Profile

The following profile shall be used:

HTTP version 1.1 or HTTP/2 shall be used. ADMF implementations shall support both.

Where used, HTTP version 1.1 shall be used as per IETF RFC 7230 [13] and related specifications.

NOTE: HTTP/1.1 defaults to the use of "persistent connections" (see IETF RFC 7230 [13], section 6.3). Implementers are encouraged to support the use of persistent connections.

Where used, HTTP/2 shall be used as per IETF RFC 7540 [21] and related specifications.

HTTP/1.1 Pipelining shall not be used.

A Requester may issue multiple HTTP requests in parallel over multiple HTTP connections or multiplexed HTTP/2 requests. However, such implementations should be aware that there is no guarantee of the order in which these requests are processed by the Responder. If such ordering is important to the Requester, it is responsible for ensuring the requests are sent out in the correct order, and for waiting for the response to each request before issuing the next one. Transfer Coding shall not be applied to the HTTP Request or Response (see IETF RFC 7230 [13], section 4).

By default, port 443 shall be used. If this is already in use, then the NE and ADMF shall be able to be configured with a port number, which shall be agreed prior to use of the standard.

By default, the ADMF shall send the HTTP requests with the path set to "/X1/NE" and the NE shall send the HTTP requests with the path set to "/X1/ADMF". An exception to the default shall only be made with strict agreement between NE and ADMF; however, implementers shall ensure that an X1 implementation can be configured with a different path if required.

8 Security

8.1 Overview

This clause details security measures to be implemented for the X1 interface. Other security aspects related to the NE (e.g. secure storage of information, access control) are out of scope of the present document.

8.2 Transport Security

8.2.1 Summary

TLS shall be used which provides authentication and authorization, integrity and confidentiality as well as replay protection between the TLS endpoints.

8.2.2 Profile

TLS shall be followed, using at least version 1.2 as defined in IETF RFC 5246 [14], supporting the recommendations given in IETF RFC 7525 [16].

New implementations should support TLS 1.3 as defined in IETF RFC 8446 [20].

8.2.3 Key generation, deployment and storage

Apart from requirements given in clauses 8.2.1, 8.2.2 and 8.2.4, aspects concerning the generation, distribution, storage and revocation of key material and certificates are out of scope of the present document. Implementations are encouraged to support best practice e.g. the guidance given in OWASP [i.1] TLS Cheat Sheet [i.1], section 2.6.

NOTE: It is assumed that the NE and ADMF are in a physically secure environment. For future uses (e.g. NFV), then this assumption would no longer be valid. Further details would then need to be added about the security of storage of key or certificate material e.g. TPM, Secure enclaves. See ETSI TR 103 308 [i.2], ETSI GS NFV-SEC 009 [i.3] and ETSI GS NFV-SEC 012 [i.4].

8.2.4 Authentication

Implementations shall perform mutual authentication using X.509 certificates following IETF RFC 6125 [17]. Implementations shall ensure that it is configurable which certificates are used.

An implementation shall consider the authentication procedure to have succeeded if TLS authentication succeeds and either or both of the following conditions are true:

- The certificate provided by the other party has a Subject field that contains a UID relative distinguished name (OID 0.9.2342.19200300.100.1.1, see IETF RFC 4519 [18], section 2.39) with a value equal to the relevant identifier (ADMF Identifier or NE Identifier) provided by the other party.
- The certificate provided contains a subjectAltName (SAN, see IETF RFC 5280 [29], section 4.2.1.6) of type uniformResourceIdentifier containing the correct certificate binding URN for the message as described in annex G.

If a Responder receives an X1 message where the authentication procedure is considered to have failed, it shall respond with an X1 error message indicating that the Requester is not authorized. If the Requester receives a response where the authentication procedure is considered to have failed, then it shall disregard the response and log an appropriate error message.

NOTE: Implementers should be aware that this mechanism requires both the Certificate Authority and TLS implementations to support use of either the UID in the Subject field or URNs in the subjectAltName field.

8.3 Additional security measures (beyond transport layer)

It will be important to follow general security best practice (e.g. use of firewalls and/or access lists to prevent denial-of-service attacks). This is out of scope of the present document. However, implementers are specifically encouraged to follow XML best practices outlined in the OWASP [i.1] XML Security Cheat Sheet [i.5].

The present document does not recommend that message-layer encryption or message-level message authentication codes are used in addition to the provisions in this clause. Of course, there may be threat models in which additional encryption may be thought to be useful. The present document does not forbid adding message-layer encryption e.g. by encrypting the whole of the payloads of the request and response messages. The details of the changes needed to do this are outside the scope of the present document.

Annex A (normative): Requirements

A.1 Basic requirements

A.1.1 Existing standards

The interface should use already existing mechanisms and standards if possible:

R1) **Future proof:** Changes can be made and new features can be added. A version structure will allow for co-existence of different versions.

R2) **Open structure:** The interface will have an open structure that will allow for extensions. Though it should be as strict as possible to make implementations as interoperable as possible. Extensions should not have any negative impact on security and other requirements.

R3) **Security:** Authentication, integrity protection and confidentiality shall be supported from end to end.

R4) **Authenticity:** The authenticity of a message can be checked in a standalone environment (e.g. no connection to an online server needed, root certificate can be enough).

R5) **Legal framework:** The present document contains a technical specification which is independent of national legislation. It does not supersede national legislation or approved practices.

R6) **Direct delivery:** Some network elements support direct delivery of IRI and CC without any additional mediation and delivery function. The interface should also support administration of these network elements.

R7) **Core functionality:** It shall be possible to provision (create, modify and delete) interceptions including all necessary parameters (e.g. CC/IRI-destination) on network nodes. It shall be possible to retrieve details of a single or all interceptions provisioned on a network node.

R8) **Administration:** It shall be possible to administrate LI relevant configuration on network nodes (e.g. update of security certificates).

A.2 Protocol & Architecture requirements

The following protocol and architecture requirements are listed:

R9) **Node Scope:** The X1 architecture and protocol shall support administration of all nodes involved in capture and control of target intercept traffic including intercept nodes and mediation and delivery functions. This shall include both on-switch and off switch probe scenarios.

R10) **Basic functionality:** The basic message exchange protocol shall be able to carry both generic LI parameters (e.g. those obtained from X1 E-warrant interface) and Interception Node manufacturer specific parameters.

R11) **Extensible:** The basic message exchange protocol shall allow limited extensibility to support parameter not currently supported by the base protocol. This extensibility shall be limited to encourage future extension of the standardized basic functionality in future versions of the X1 standard.

R12) **Flexibility:** The X1 architecture and message exchange technique shall be flexible to allow implementation in both existing and future national and international operator network architectures. As a minimum it shall be compatible with 3GPP, TISPAN/NTECH, NFV SEC, ETSI TC LI, ANSI and other international network architecture and handover standards.

R13) **One-to-many:** The architecture and message protocol shall support both one-to-one and one-to-many LI end point configurations (i.e. it shall be possible to provision hundreds of end points simultaneously and efficiently).

R14) **Backwards compatibility:** The X1 architecture and protocol shall be backwards compatible with existing LI devices where possible. Specifically the standardized X1 shall not place significantly more performance or load impacts than existing proprietary approaches on LI nodes.

There is no specific requirement to retro-fit this X1 standard onto existing IP or legacy circuit switched nodes, although the standard does not prohibit such retrofitting where practical. Parallel running of X1 and legacy or proprietary interfaces shall be supported where practical. The X1 architecture shall permit different versions of X1 to be running on different components and (as far as is practical) the functionality from the older version shall still continue to work (though features introduced in the new versions shall cause errors to be sent).

R15) **Lightweight:** Many LI devices (e.g. Switches/Routers) currently use lightweight protocols such as SNMP, and have limited processing power and/or limited application layer intelligence. The protocol shall be designed to support such lightweight devices.

R16) **Permanent and dynamic connections:** The X1 architecture and message exchange technique shall support both permanent connection and dynamic link/connection scenarios.

R17) **Direct delivery:** Support situation where interception is delivered direct to LEMF without further CSP mediation. No need to explicitly draw this out but do allow enough information over X1 to support this situation.

R18) **Delay:** The X1 architecture and message exchange technique shall by design not introduce undue delay compared with existing proprietary X1 implementations.

R19) **Dynamic Triggering and HI1:** The X1 architecture and message exchange technique shall be compatible and interoperable with both ETSI TC LI HI1 and Dynamic Triggering standards.

A.3 Security requirements

R20) **Authentication:** The X1 architecture and message exchange technique shall provide both authentication of physical end points and authentication of the software application receiving the message.

NOTE: Requirement is limited to authenticating the LI function identity and not authenticating the software version or integrity.

R21) **Authorization:** The X1 architecture and message exchange technique shall provide both authorization of physical end points and authorization of the software application receiving the message.

R22) **Accounting and audit:** The X1 architecture and message exchange technique shall include sufficient information to enable Accounting & Auditing functions in the ADMF and NE.

R23) **Integrity protection:** The X1 message exchange technique shall provide integrity protection for all messages exchanged between nodes in the X1 architecture. Use of Integrity protection shall be mandatory.

R24) **Confidentiality protection:** The X1 message exchange technique shall provide confidentiality protection for all messages exchanged between nodes in the X1 architecture.

R25) **Replay protection:** The X1 message exchange technique shall provide replay protection for all messages exchanged between nodes in the X1 architecture.

R26) **Standalone interface:** The X1 architecture and message exchange technique shall be designed as a standalone physically dedicated LI interface. The design and selection of the protocol shall where possible ensure vulnerabilities in non-LI interfaces on the same node shall not impact LI interfaces and security.

R27) **Hardened Protocol:** The X1 message exchange technique shall use a harden protocol containing minimal options or extensions which are not specifically required by X1.

R28) **Minimum Security Level:** The X1 architecture and message exchange techniques shall provide a minimum level of security (including cypher suites and key length), which shall be supported by all nodes. At least two algorithms shall be specified. The protocol and algorithms shall be resistant to bid down attack.

R29) Underlying Infrastructure Trust: The X1 architecture and message exchange techniques shall assume by default that the underlying network communication links and infrastructure are untrusted.

R30) Firewall and NAT Transversal: The X1 message exchange technique shall be compatible with existing operator firewall and NAT transversal architectures. The message exchange technique shall not require unrestricted opening of common ports (e.g. port 80 or 21). The message exchange technique shall not prohibit the development of future X1 aware firewall filtering to provide rejection of malicious X1 message at operator security gateways.

R31) Certificate and Key Management: The X1 architecture and message exchange techniques shall include (where applicable) Certificate and Key Management mechanisms. In addition mechanisms for Certificate/Key revocation shall be provided.

R32) Single Node Compromise: The X1 architecture and message exchange techniques shall ensure that a vulnerability or weak implementation in one node does not adversely affect other nodes. Specifically it shall not be possible to attack one interception node by using recovered plan text or other security parameters from a vulnerable one.

R33) Node Administration: The X1 architecture and message exchange techniques shall ensure by design that within node implementations, non-LI super-users can be prevented from making LI related parameters changes without authority from and knowledge of the LI administrator.

R34) Encryption of target information: It shall be possible to use encrypted target information only by use of encrypted targets and encryption keys. In case of encrypted information it shall be possible to change encrypted target information and encryption keys periodically without interruption of any active interception.

A.4 Other requirements

A.4.1 Performance statistics (for further study)

Performance requirements include:

- In general or per LI measure.
- Activity: Amount of intercepted traffic? Maximum and average bandwidth? Minutes of intercepted voice? Count of intercepted messages? Time of last activity?
- Maximum number of parallel interceptions (e.g. in busy hours).
- Maximum number of parallel intercepted accounts/connections with same target identifier (e.g. in case of IMEI duplicates).

The performance requirements are derived from measures of the amount and rate of Lawful Interception. Clearly this will vary but some guidelines are as follows:

- Considerations of the bandwidth of intercepted traffic are in general not relevant to X1 (except perhaps for a NE to report that bandwidth is exceeding certain parameters).
- Number of targets on cover at any given time:
  - This number is usually very small compared to the total number of users and for the purposes of the present document will be considered as tens or hundreds at most.
- Are there situations where a single target on cover causes a lot of X1 messages. Consider the following ways this could happen:
  - Can a single target cause a large number of target identifiers to be tasked (consider roaming)?
  - Can one have a large number of HI1 messages for each target identifier (frequent changing of parameters)?

- For a single ADMF-NE link, can one have lots of X1 messages for a given HI1 message arriving at the ADMF?
- How many different NEs can each ADMF have to talk to?

A.4.2 Capability detection

Automatic capability detection is not covered in the present document.

A.4.3 Remote triggering

Remote triggering is defined as a system where a trustworthy node contains the target list. Instead of maintaining a list of intercepted targets on a (less trustworthy) network node, the start of all communication (calls, data session, etc.) could be reported to another (trustworthy) node which checks for intercepted targets and dynamically triggers interceptions on the first node.

Remote triggering is not covered in the present document.

A.4.4 Requirements to be handled by the transport layer

R35) Ability to send frequent messages from ADMF to NE to add/delete, with an OK/not OK response.

R36) Ability to send frequent list messages, with a status update response.

R37) Ability to send occasional urgent messages from NE as error messages, with a "received OK" response.

R38) Reliable transport - need to know if message failed to get through.

R39) Able to be secured using standard techniques. Discuss whether there are concerns about what has to be opened in various firewalls to let it through.

R40) Simple and lightweight, suitable for use on standard network equipment in broadband (e.g. router) and mobile communications (e.g. SGSN).

R41) Helpful (non-essential) if it is able to group multiple messages together so that one security check is not needed for each message (this can be handled by a grouping function within our message layer though nicer not to).

R42) No unnecessary buffering or delays of some messages compared to others, though perhaps does not need to guarantee the order of delivery of messages.

R43) No QoS - the interface will not prioritize or buffer any information. Needs to deliver messages to end point, which can either accept the message (and buffer/prioritize if it chooses) or reject.

R44) Every message requires a response:

- Helpful if it can relay an immediate "*don't understand*" response as a reply to a message i.e. without understanding its contents.
- Need to be able to respond quickly with errors e.g. parsing errors.
- Need to be able to respond quickly with an OK message.

No messages to be stalled/buffered or rejected by the transport layer because the receiving application layer is busy creating a response.

Annex B (normative): Use of extensions

B.1 Overview

The present document defines a number of extension points, including in the TaskDetails structure (see clause 6.2.1.2), and TargetIdentifier format (see table 6.2.1.2-2). This clause defines how extensions are to be used in table 6.2.1.2-1 and table 6.2.1.2-2.

B.2 Extension definitions

Where a feature or information element already exists in the present document, it shall be used in preference to any extended field. Extensions shall not be drafted as an alternative or re-formatting of functionality or information that already exists within the present document.

An extension shall be a structure (e.g. a complexType in XSD) defined in a separate schema, and shall contain at a minimum the following elements.

Table B.2-1: Extension fields

Field	Description	Format	M/C/O
Owner	Human-readable indication of the entity responsible for the definition and maintenance of the extension.	UTF-8 string.	M

The extensions shall be defined in a namespace belonging to the entity responsible for drafting and maintaining the extension. It shall not be defined in the namespace of the present document.

Annex C (normative): Using Task Object at Mediation and Delivery Functions

C.1 Overview

An ADMF may use X1 messages to provision a mediation and delivery function instead of a point of interception, following the deployment model given in clause 4.1.4. This annex describes how the usage and meaning of the messages defined in clause 6 differ when used for this purpose. Unless otherwise specified, the messages are used as for any other NE.

C.2 TaskDetails

C.2.1 General

The TaskDetails structure used in the ActivateTask and ModifyTask messages are used as for an NE with the differences described in the following clauses.

C.2.2 MediationDetails structure

The MediationDetails structure provides additional details for a Task, specific to Mediation and Delivery Functions. Multiple instances of the MediationDetails structure may be used to indicate that multiple LIIDs are associated with the task.

When a ModifyTask message is received by the MDF from the ADMF, the MDF shall, upon successful processing and execution of the ModifyTask message, ensure that:

1) only the LIIDs included in the ModifyTask message (via a MediationDetails structure) remain active; and

2) any LIIDs that were associated with the task identified in the ModifyTask message, but were not identified in the ModifyTask message, shall be deactivated (i.e. those intercepts shall cease).

To clarify the above, suppose that TaskID A had LIID 4 and LIID 5 associated with it and interception was active on both LIID 4 and LIID 5. If a ModifyTask message is received and successfully processed by the MDF with a single MediationDetails structure that includes LIID 4, then the interception on LIID 4 will remain active while the interception on LIID 5 will cease.

Table C.2.2-1: Mediation Details structure

Field	Description	Format	M/C/O
LIID	Lawful Interception Identifier associated with the Task.	LIID as defined in ETSI TS 103 280 [4].	M
DeliveryType	Statement of whether to deliver HI2 and/or HI3 for this LIID.	Enumerated value - one of "HI2Only", "HI3Only" or "HI2andHI3".	M
StartTime	Start time associated with the activation of interception or mediation for the Task (which may be in the future).	Timestamp.	O
EndTime	End time associated with the deactivation of interception or mediation for the Task.	Timestamp.	O
ListOfDIDs	Details of where to send the intercepted traffic for this LIID. Shall be included if deviation from the taskDetails ListofDIDs is necessary. If included, the details shall be used instead of any delivery destinations specified in the ListOfDIDs field in the TaskDetails structure.	List of Destination Identifiers (DID) referencing the desired delivery destination records.	C

Field	Description	Format	M/C/O
MediationDetailsExtension	One or more extension placeholders; each may be populated by a list of elements defined by external specifications.	See annex B.	O
ServiceScopingOptions	Shall be included to Identify the service(s) and associated service-related delivery settings for this LIID if there is no default setting in the MDF for ServiceScoping or to override the default setting in the MDF for ServiceScoping. May include more than one instance of this parameter to allow for different combinations of sub-parameters associated with a single LIID. This parameter is defined in more detail in table C.2.2-2.	Sequence containing - one or more instances of the ServiceScoping options listed in table C.2.2-2.	C
ListOfTrafficPolicyReferences	Ordered list of TrafficPolicyReferences to be applied to the LITaskObject. Shall be included if deviation from the taskDetails ListOfTrafficPolicyReferences is necessary. If included, the details shall be used instead of any traffic policies specified in the ListOfTrafficPolicyReferences field in the TaskDetails structure.	Given in ETSI TS 103 120 [28], clause 8.2.13 ListOfTrafficPolicyReferences.	C

Table C.2.2-2: Service Scoping structure

Field	Description	Format	M/C/O
ServiceType	Shall be included to Identify the service(s) to be reported for this LIID per the description beneath the table. The values given in this field indicate the services to which the other options in the Service Scoping structure shall apply.	One or more of the following enumerated values: • "voice". • "data". • "messaging". • "pushToTalk". • "LALS". • "RCS".	C
LocationType	Shall be included to Identify whether and under what conditions to deliver location information for this LIID per the description beneath the table.	Choice of: • "doNotReport". • One or more of the following enumerated values: – "reportBeginningAndEnd". – "reportUponChange". – "reportLALS".	C
SuspendOnOutboundInternationalRoaming	Shall be included to Identify whether to suspend interception or not (i.e. continue interception) if the target undergoes outbound international roaming and per the description beneath the table.	Boolean.	C
ReportPostDialledDigits	Shall be included to identify whether reportPostdialleddigits are to be reported for this LIID for the ServiceType of "voice" per the description beneath the table. Not applicable to other service types.	Boolean.	C

For ServiceType, LocationType, SuspendOnOutboundInternationalRoaming, or ReportPostDialledDigits, If there is no default setting in the MDF or to override the default setting in the MDF, the corresponding parameter shall be included.

NOTE: RCS is defined as Rich Communication Services (see GSMA RCC.07 [i.6]).

The ADMF and MDF shall support the end time signalling, a means by which the ADMF is able to provide the MDF with the EndTime parameter and where the MDF applies the EndTime parameter to MDF based task activity operations. When the end time signalling is to be used for a task, the ADMF shall include within the MediationDetails the EndTime parameter with a value based on the end time in the warrant for a task and shall also include the "ImplicitDeactivationAllowed" flag in the TaskDetails. If the MDF determines that the EndTime has been reached for a task, the MDF shall deactivate the associated task and notify the ADMF of this implicit deactivation via a ReportTaskIssue message with a TaskReportType of "ImplicitDeactivationAllowed". In this case, the MDF shall consider that it was alternatively configured by the CSP (see clause 6.6.2) with respect to the keepalive procedure (i.e. the MDF shall not deactivate the associated task based on the expiration of timer P2).

Annex D (normative): Hashed Identifiers

D.1 Overview

Hashed identifiers provide an alternative to providing plain-text target identifiers over X1. This is intended to provide a measure of additional security against disclosure of such target identifiers. However, it should be noted that this technique does not provide protection against:

- An attacker in possession of hash information from verifying whether a specific given identifier matches a given hash or salt.
- An attacker in possession of complete hash information (including salt) from recovering identifiers that have a small set of possible values (e.g. MSISDN numbers in a particular country) by brute force attack.

Instead, this technique is intended to provide a simple extra layer of protection against e.g. accidental disclosure via a user interface.

D.2 Hashed Identifier Usage

D.2.1 Overview

An ADMF wishing to provision an NE with a hashed identifier uses the following procedure:

1) The ADMF populates a Hash Context object with the operator's chosen hash algorithm identifier and a random salt value (see clause D.2.2).
2) The ADMF issues a CreateObject request containing the Hash Context object to the NE (see clause 6.8.1).
3) The ADMF calculates the hash digest of the required plain-text identifier using the details from the Hash Context (see clause D.2.3.2).
4) The ADMF populates a HashedIdentifier structure with the digest, along with an indication of the target identifier type and the identifier of the Hash Context object containing the salt (see clause D.2.3).
5) The ADMF issues an ActivateTask request containing the HashedIdentifier to the NE (see clause 6.2.1).

The NE can now inspect each candidate identity and create a hash digest using the information in the Hash Context. If the digest matches the one in the HashedIdentifier structure, the NE can consider the target identity to have matched.

Hashed Identifiers may only be used for target identifier types which derive from simple types such as xs:token, and which specify a single unambiguous value as a target identifier. Hashed Identifiers may not be used for:

- target identifier types which are complex types due to potential ambiguities in forming a canonical binary representation (see clause D.2.3);
- target identifier types which do not describe a single unambiguous value (such as tcpPortRange) since it is impossible to determine whether a given identifier matches the target identifier by comparing hashes.

However, a given Task may contain both hashed and non-hashed target identifiers (e.g. a hashed IPv4 address along with a plain-text tcpPortRange) in its targetIdentifiers list.

D.2.2 Hash Context

A Hash Context is derived from a Generic Object (see clause 6.8.1.1) and consists of the following elements:

Table D.2.2-1: Hash Context structure

Field	Description	Format	M/C/O
hashAlgorithm	Gives the object identifier of the hash context containing the relevant configuration details used to calculate the hash digest.	Hash name string matching one of those defined in the IANA Named Information Hash Algorithm Registry [25].	M
salt	Salt to be used when calculating the hash digest value (see clause D.2.3). Shall be at least 8 octets long.	XML hexbinary representation of the salt value.	M

The choice of hash algorithm is made by the operator and enforced by the ADMF. ADMFs and NEs supporting hashed identifiers shall support the use of the following hash algorithms:

- sha-256 with 256-bit value length as defined in IETF RFC 6920 [25].
- sha-512 with 512-bit value length as defined in IETF RFC 6920 [25].
- sha3-512 with 512-bit value length as defined in FIPS PUB 202 [26].

Additional algorithms may be supported in both the ADMF and NE.

If the ADMF requests the creation of a Hash Context object with an unsupported hash algorithm or an insufficiently long salt, the NE shall reject the request with an appropriate error.

A Hash Context and its associated salt may be used by multiple HashedIdentifier instances (see clause D.2.3) to reduce the processing burden at the NE, at the cost of reducing the number of salts that an attacker would need to deal with if attempting to exhaustively search for the original target identifier.

D.2.3 HashedIdentifier

D.2.3.1 Structure

A HashedIdentifier consists of the following elements.

Table D.2.3.1-1: HashedIdentifier structure

Field	Description	Format	M/C/O
hashContextID	Gives the object identifier of the hash context containing the relevant configuration details used to calculate the hash digest.	X1ObjectID (see clause D.2.2).	M
targetIdentityType	Name of the equivalent plain-text target identity element from the TargetIdentifier type. Only simple types are supported.	String.	M
hashDigest	Digest of the target identifier and salt.	XML hexbinary representation of the binary digest.	M

D.2.3.2 Hashing procedure

It is essential that both the ADMF and NE calculate the hashDigest value in the same way. The hashDigest value shall be calculated according to the following procedure at both NE and ADMF:

1) Ensure that any plain-text target identity used to calculate a hashDigest is first correctly normalized into the format defined by the relevant TargetIdentifier format (see table 6.2.1.2-2).

2) For values where value comparisons are case-insensitive, transform the plain-text identity to lower-case. In cases where parts of the value are case-insensitive and others are not (e.g. SIP URI) then only the case-insensitive parts shall be lower-cased.

3) Obtain a binary representation of the plain-text target identity:

   - For simple types derived from xs:token or xs:string, the binary representation shall be the octets giving the UTF-8 encoding of the plain-text string.

   - For simple types derived from xs:hexbinary, the binary representation shall be the octets represented by the hexbinary notation.

   - For simple types derived from xs:integer and which represent unsigned numbers, the binary representation shall be the octets of the binary representation of that number given in network byte order (i.e. big endian).

4) Concatenate the octets of the salt value from the associated Hash Context to the end of the binary representation of the identity.

5) Take the hash of the concatenated result using the hash algorithm identified by the associated Hash Context.

D.3 Worked examples

D.3.1 Worked example 1

D.3.1.1 Initial information

Table D.3.1.1-1: Initial information

Information element	Value
Chosen hashing algorithm	sha-256
Plain-text target identity type	InternationalE164
Plain-text target identity value	"447700900000"

D.3.1.2 Construction of the Hash Context

The ADMF chooses a salt value of 0x4241792fc4d3d097, and allocates a random UUID for an object identifier. The ADMF now has enough information to populate a Hash Context object.

```
<X1Object xsi:type="HashContext">
  <x1ObjectID>e3d62e2b-d211-433d-b0f9-488ed89ba7c0</x1ObjectID>
  <hashAlgorithm>sha-256</hashAlgorithm>
  <salt>4241792fc4d3d097</salt>
</X1Object>
```

The ADMF may now issue a CreateObjectRequest message to the NE with this Hash Object (see clause 6.8.1).

D.3.1.3 Binary representation of the target identity

In this case, the identity is of type InternationalE164. This is a type derived from xs:token, so the binary representation is the bytes of the utf-8 representation of the string.

```
binary_representation = 0x343437373030393030303030
```

D.3.1.4 Concatenation with the salt

The salt is taken from the Hash Context that the ADMF wishes to use, and is appended to the end of the binary representation.

```
concatenated_value = binary_representation || salt
     = 0x34343737303039303030303041241792fc4d3d097
```

D.3.1.5 Calculation of the hash digest

The ADMF can now calculate the hash digest of the concatenated value, using the hash algorithm identified in the Hash Context object (in this case, sha-256).

```
hashDigest = sha-256(concatenated_value)
    = sha-256(0x34343737303039303030303041241792fc4d3d097)
= 0xddbe522009b5b32f1b84c82c06dedc0d24ba373d4ae244790fd071076b4536c0
```

D.3.1.6 Construction of the HashedIdentifier

The ADMF now has all the information it needs to create a HashedIdentifier target identity for use in an ActivateTask message towards the NE.

```
<hashedIdentifier>
  <hashContextId>e3d62e2b-d211-433d-b0f9-488ed89ba7c0</hashContextId>
  <targetIdentityType>InternationalE164</targetIdentityType>
  <hashDigest>ddbe522009b5b32f1b84c82c06dedc0d24ba373d4ae244790fd071076b4536c0</hashDigest>
</hashedIdentifier>
```

Annex E (normative): Destination Sets

E.1 Overview

When intercepted traffic is to be delivered by the NE to a Destination which belongs to a group of related Destinations DIDs can be grouped together under a single DSID (see clause 5.1.3).

When Destination Sets are used each Task is associated with one or more Destination Sets. Prior to associating a Task with a given DSID, it is required that a Destination Set with the DSID has already been created as described in clause E.2 but there is no requirement that a connection has been successfully established for that DSID.

Checks regarding availability and status of downstream delivery of information are outside the scope of the present document.

E.2 Destination Set Usage

E.2.1 Overview

All Generic Object Methods are applicable to DestinationSetDetails Objects.

An ADMF wishing to use a DSID within a provisioning request towards an NE uses the following procedure:

- The ADMF populates a DestinationSetDetails object with the identifiers and values as described in clause E.2.2.
- The ADMF issues a CreateObject request, containing the DestinationSetDetails object, to the NE (see clause 6.8.1).
- The ADMF issues an ActivateTask request containing the DestinationSetDetails Generic Object ID(s), also referred to as the DSID(s), to be used within the ListofDIDs field (see clause 6.2.1).

It is required that a Destination with the DID has already been created (as described in clause 6.3) before it can be included within a Destination Set Details object, although there is no requirement that a connection has been successfully established for that DID.

When a Destination Set is created or modified, the ADMF shall ensure that the resulting Destination Set includes at least one Destination and shall not issue a request that would result in an empty DestinationSet at the NE.

If the NE receives a request to create/modify a Destination Set resulting in an empty DestinationSet, it shall reject the request with an appropriate error response.

When a Destination Set is created or modified, all the involved Destinations shall have compatible delivery types, i.e. at least one common delivery type (e.g. all destinations have DeliveryType "X2Only" or "X2andX3"). The ADMF shall ensure that a DestinationSet with incompatible delivery types is not created and shall not issue a request that would result in the creation of such a DestinationSet to the NE.

If the NE receives a request to create/modify a Destination Set including Destinations with incompatible DeliveryTypes, it shall reject the request with an appropriate error response.

NOTE: The NE may store an information about the DeliveryType resulting from the inclusion of Destinations into the Destination Set, for immediate check when a task is created/modified for a given Destination Set.

Depending on the NE implementation, it may not be possible to modify some elements or all of a Destination Set details while the Destination Set is in use. If the NE cannot modify one or more of the elements in the ModifyObject request, it shall reject the entire ModifyObject request with an appropriate error response.

The length of time an NE requires to make the changes requested in the ModifyObject request message is an implementation detail, but the expectation is that changes are made without undue delay.

A Destination Set may only be removed if it is not referenced by any Tasks. An NE shall respond with an appropriate error if the ADMF attempts to remove a Destination Set that is referenced by a Task.

E.2.2 DestinationSetDetails Object

A DestinationSetDetails object consists of the following elements.

Table E.2.2-1: DestinationSetDetails structure

Field	Description	Format	M/C/O
FriendlyName	A human-readable name associated with the Destination Set.	Free-text string.	O
ListOfAssociatedDIDs	One or more AssociatedDID structures.	As defined in table E.2.2-2.	M
DestinationSetDetailsExtensions	One or more extension placeholders; each may be populated by a list of elements defined by external specifications.	See annex B.	O
DestinationSetType	Shall be included to identify how IRI and/or CC should be distributed across the DIDs within the Destination Set.	Enumerated value - one of "Redundant" or "Duplicate".	M

The AssociatedDID structure is defined as follows.

Table E.2.2-2: AssociatedDID

Field	Description	Format	M/C/O
DID	See clause 5.1.	See clause 5.1.	M
Preference	Where the DestinationSetType is "Redundant" the preference value is an integer representing the DIDs preference of use within the Destination Set. Where the DestinationSetType is "Duplicate" the preference value shall be "0".	Integer.	M

Where the DestinationSetType included within the DestinationSetDetails is "Redundant" the POI will use the specified DIDs as a set of redundant end points, it is mandatory for the "Preference" to be defined for each DID within a Destination Set where the DestinationSetType is "Redundant". Preference defines the DIDs order of use with the smallest integer indicating the most preferred DID(s). Should the most preferred DID(s) become unavailable the next preferred and available DID(s) shall be used.

It is an implementation decision for the NE to determine whether to duplicate traffic if two or more DIDs with the same Preference value are referenced within the same DestinationSetDetails object.

Where the DestinationSetType included within the DestinationSetDetails is "Duplicate", the NE will send copies of intercepted traffic to all DIDs within the Destination Set.

Annex F (normative): Traffic Policies and IRI Policies

F.1 Overview

This annex describes how Traffic Policy information (as defined in ETSI TS 103 120 [28], clause 7.5) and IRI Policy information (as defined in ETSI TS 103 120 [28], clause 7.7) can be described and transported within X1 messages. This supports deployments where Traffic Policy or IRI Policy information is required to be shared with an NE. Decisions on which functions Traffic Policy or IRI Policy information are required to be sent to are for agreement between the LEA and CSP, and are out of scope of the present document. An ADMF is not required to have an HI-1 interface compliant with ETSI TS 103 120 [28] in order to use the capabilities described in this annex, but such a deployment is supported.

F.2 Traffic Policy Usage

F.2.1 Overview

ETSI TS 103 120 [28] defines a set of Traffic Policy objects in clause 7.5 and in the XSD definitions attached to that specification. The present document imports those definitions directly in order to maximize interoperability.

Specifically, Traffic Policies are associated to Tasks by the ListOfTrafficPolicyReferences in the TaskDetails structure as defined in clause 6.2.1.2, table 6.2.1.2-1, and by the ListOfTrafficPolicyReferences in the Mediation Details structure as defined in table C.2.2-1.

F.2.2 Traffic Policy Object

A Traffic Policy Generic Object contains the fields defined in ETSI TS 103 120 [28], clause 7.5 with the following clarifications.

The ObjectIdentifier field in each TrafficRuleReference (see ETSI TS 103 120 [28], table 7.22) shall be interpreted as a Generic Object ID (see clause 6.8.1.3).

F.2.3 Traffic Rule Object

The Traffic Rule Generic Object contains the fields defined in ETSI TS 103 120 [28], clause 7.6.

F.3 IRI Policy Usage

F.3.1 Overview

ETSI TS 103 120 [28] defines a set of IRI Policy objects in clause 7.7 and in the XSD definitions attached to that specification. The present document imports those definitions directly in order to maximize interoperability.

If the IRI from specific types of NFs or specific IRI events should not be generated or delivered, the IRI Policies containing the IRI criteria should be associated to the relevant Tasks by the ListOfIRIPolicyReferences in the TaskDetails structure as defined in clause 6.2.1.2, table 4, and by the ListOfIRIPolicyReferences in the Mediation Details structure as defined in table C.1.

F.3.2 IRI Policy Object

An IRI Policy Generic Object contains the fields defined in ETSI TS 103 120 [28], clause 7.7 with the following clarifications.

The ObjectIdentifier field in each IRIRuleReference (see ETSI TS 103 120 [28], table 7.27) shall be interpreted as a Generic Object ID (see clause 6.8.1.3).

F.3.3 IRI Rule Object

The IRI Rule Generic Object contains the fields defined in ETSI TS 103 120 [28], clause 7.8.

Annex G (normative): Certificate binding URN

G.1 Overview

A certificate binding URN is a URN value under the ETSI TC LI namespace root `urn:etsi:li`.

It provides a value that can be carried as a subjectAltName and used to bind a client or server certificate to a specific identifier and role (NE or ADMF), and to ensure that a certificate is intended to be used for X1.

G.2 URN format

A certificate binding URN has the following format, with the placeholders `{role}` and `{identifier}` given as per table G.2-1.

```
urn:etsi:li:103221-1:cert-binding:{role}:{identifier}
```

Table G.2-1: Binding URN values

Field	Description	Format
Role	String indicating the role of the party presenting the certificate.	One of "ADMF" or "NE".
Identifier	String giving the relevant identifier of the party presenting the certificate (ADMF Identifier if the role is ADMF, otherwise NE Identifier).	String containing a value as per table 6.1-1.

G.3 Validity

A certificate binding URN presented in a client or server certificate as part of the transmission of an X1 message shall be considered valid if and only if all the following conditions are met:

- It follows the format given in clause G.2.
- The Role value correctly matches the expected role of the presenting party (i.e. ADMF or NE).
- The Identifier value correctly matches the relevant identifier in the X1 message (i.e. the ADMF identifier if the Role is given as "ADMF", or the NE Identifier if the Role is given as "NE").

Annex H (normative): Configuration Information

H.1 Overview

This annex is only applicable when the X0 interface (see ETSI TS 104 000 [30]) is used to configure the X1 interface.

The ADMF may provide X1 configuration to the NE as part of X0 operations (see clause 6.2.5.2 in ETSI TS 104 000 [30]) using the X1ConfigurationDetails structure given in clause H.2.

A schema for the X1ConfigurationDetails structure is given as part of the XSD schema provided as an attachment to the present document.

H.2 X1ConfigurationDetails

Table H.2-1: X1ConfigurationDetails

Field	Description	Format	M/C/O
neIdentifier	NE identifier, uniquely identifies the NE to the ADMF (see table 6.1-1 in clause 6.1).	See table 6.1-1 in clause 6.1	M
time1RequestTimeout	TIME1 in seconds as defined by clause 5.2.2.	Time1RequestTimeout Min value=1	M
time2RequesterTimeout	TIME2 in seconds as defined by clause 5.2.2.	Time2RequesterTimeout Min value=2	M
x1KeepaliveDetails	Keepalive parameters, as defined by clause 6.6.2.	X1KeepaliveDetails (see table H.2-2)	M
deactivateAllTasksEnabled	Indicates whether deactivation of all tasks shall be enabled (see clause 6.2.4).	Boolean	M
removeAllDestinationsEnabled	Indicates whether removal of all destinations stored at the ELI shall be enabled (see clause 6.3.4).	Boolean	M
deleteAllObjectsEnabled	Indicates whether deletion of all objects shall be enabled (see clause 6.8.6).	Boolean	M

Table H.2-2: X1KeepaliveDetails

Field	Description	Format	M/C/O
keepaliveEnabled	Specifying per NE (ELI) whether Keepalives are enabled or disabled, see clause 6.6.2.	Boolean	M
keepaliveTimeP1	Value in seconds to monitor the response from ADMF to the ReportNEIssues as defined by clause 6.6.2. May be present only if keepaliveEnabled is set to "True".	KeepaliveTIMEP1 Min value=1	C
keepaliveTimeP2	Value in minutes as defined by clause 6.6.2. May be present only if keepaliveEnabled is set to "True".	KeepaliveTIMEP2 Min value=1	C
keepaliveTimeP3	Value in minutes as defined by clause 6.6.2. May be present only if keepaliveEnabled and deactivateAllTaskEnabled are set to "True". Value "0" indicates that deactivation of all tasks shall not be performed as part of the keepalive procedure.	KeepaliveTIMEP3 Min value=0	C

Annex I (informative): Tasking and operational flows

I.1 Overview

This annex provides examples of X1 message tasking flows, X2/X3 PDU operational flows, and PDU handover flows between various entities, showing the relationship between various X1 Task fields, X2/X3 PDU fields and conditional attributes and handover PDU fields.

The flow diagrams use fields such as:

- X1 XID. Uniquely identifies an X1 Task. May map to a single LIID or multiple LIIDs. See clause 5.1.2.
- X1 ProductID. Optional field of an X1 Task. If present, used by the NE to populate the X2/X3 PDU XID instead of the Task's XID. See clause 6.2.1.2, table 6.2.1.2-1, "ProductID".
- X1 LIID. See clause C.2.2, table C.2.2-1.
- X2/X3 XID field. Contains the X1 ProductID if present, otherwise the X1 XID. See ETSI TS 103 221-2 [19], clause 5.2.7.
- X2/X3 Additional XID Related Information (AXRI) conditional attribute. The PDU may contain 1 or more AXRIs. The AXRI is used if an NE needs to perform deduplication by sending a single X2 and/or X3 PDU associated with more than one XID (because more than one Task matches the PDU). See ETSI TS 103 221-2 [19], clause 5.3.22.
- Handover LIID. See ETSI TS 102 232-1 [i.7], clause 5.2.2.

NOTE: The values shown in the flow diagrams for these fields are examples only.

The flow diagrams also show the recommended order of tasking of NEs such as POIs, TFs, and MDFs for Task activation and deactivation, to minimise issues of under-collection and over-collection of information.

The flow diagrams only show one occurrence of various NE entity types (e.g. POI, TF and MDF), even though in practice multiple instances of these entities are deployed.

The flow diagrams do not describe CIN allocation nor any correlation behaviour between X2 and X3 PDUs.

I.2 Example flows

I.2.1 Single LIID

An example of tasking and operational flows for a single LIID 100 for target A is shown in figure I.2.1-1. The interception is activated before the session starts and the interception is deactivated while the session is still active.

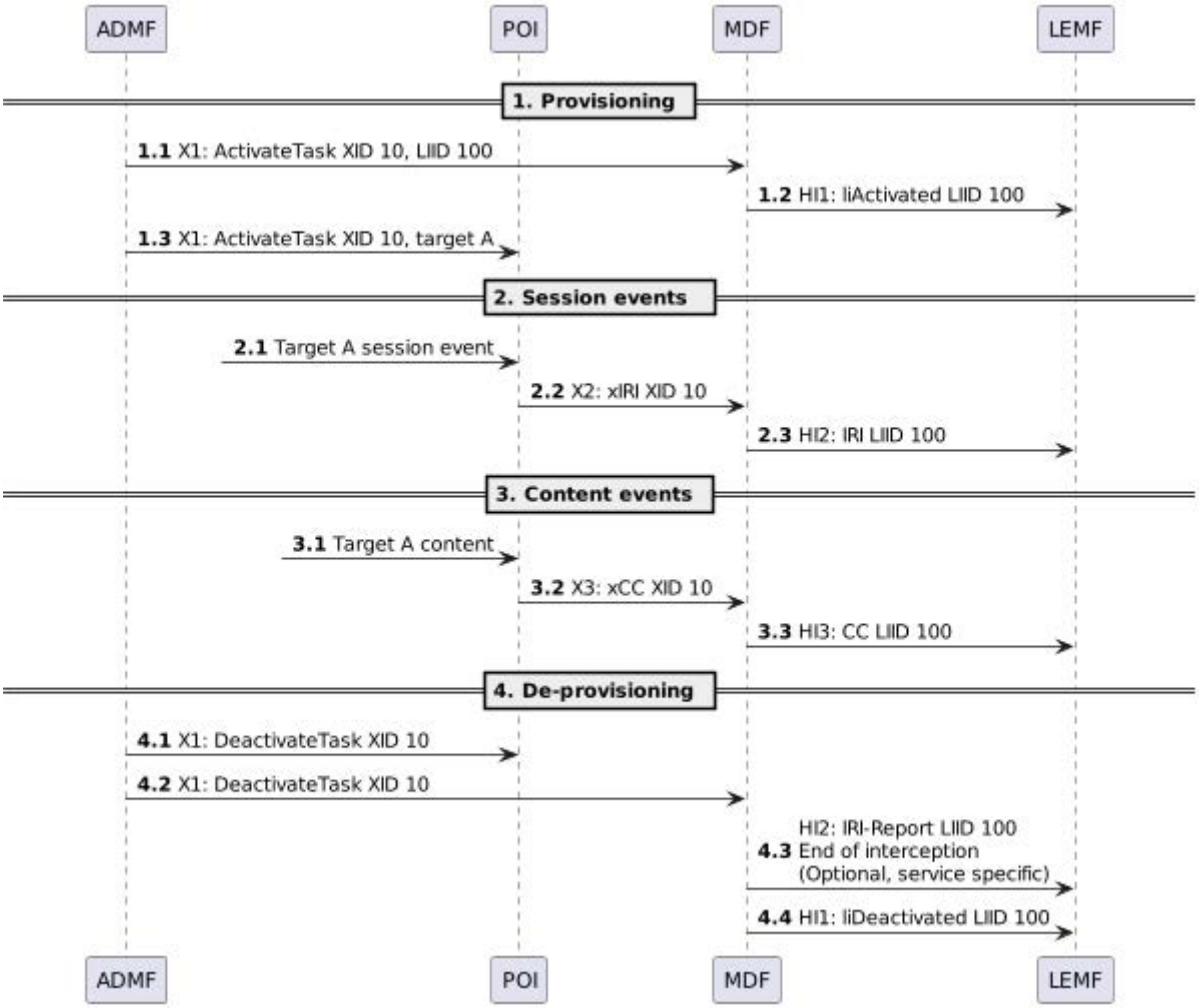

Figure I.2.1-1: Single LIID flows

The flow steps are:

1. ADMF provisions new Task:

   1.1 ADMF sends MDF an X1 ActivateTask message for XID 10, LIID 100. MDF is provisioned first to ensure that it is ready to receive X2/X3 PDUs.

   1.2 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 100.

   1.3 ADMF sends POI an X1 ActivateTask message for XID 10, target A, without an LIID.

2. Session events processed by POI:

   2.1 POI receives a session event for Target A.

   2.2 POI sends MDF an xIRI PDU (over X2) for XID 10 containing the session data.

   2.3 MDF sends LEMF an IRI PDU (over HI2) for LIID 100.

3. Content events processed by POI:

   3.1 POI receives a content event for Target A.

   3.2 POI sends MDF an xCC PDU (over X3) for XID 10 containing the content.

   3.3 MDF sends LEMF a CC PDU (over HI3) for LIID 100.

NOTE: The network session and content events may occur at any time during the lifetime of the Task.

4. ADMF deprovisions Task:

    4.1 ADMF sends POI an X1 DeactivateTask message for XID 10. POI is deprovisioned first to ensure that no more X2/X3 PDUs are sent to MDF.

    4.2 ADMF sends MDF an X1 DeactivateTask message for XID 10.

    4.3 MDF sends LEMF an IRI-Report PDU (over HI2) for end of interception, due to the active session. This is optional and service-specific.

    4.4 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 100.

I.2.2 Multiple LIIDs sharing an XID

An example of tasking and operational flows for multiple LIIDs sharing an XID for target B is shown in figure I.2.2-1. Three LIIDs are activated, the session starts, one LIID is deactivated with an active session, content is intercepted, the session ends, and the remaining LIIDs are deactivated.

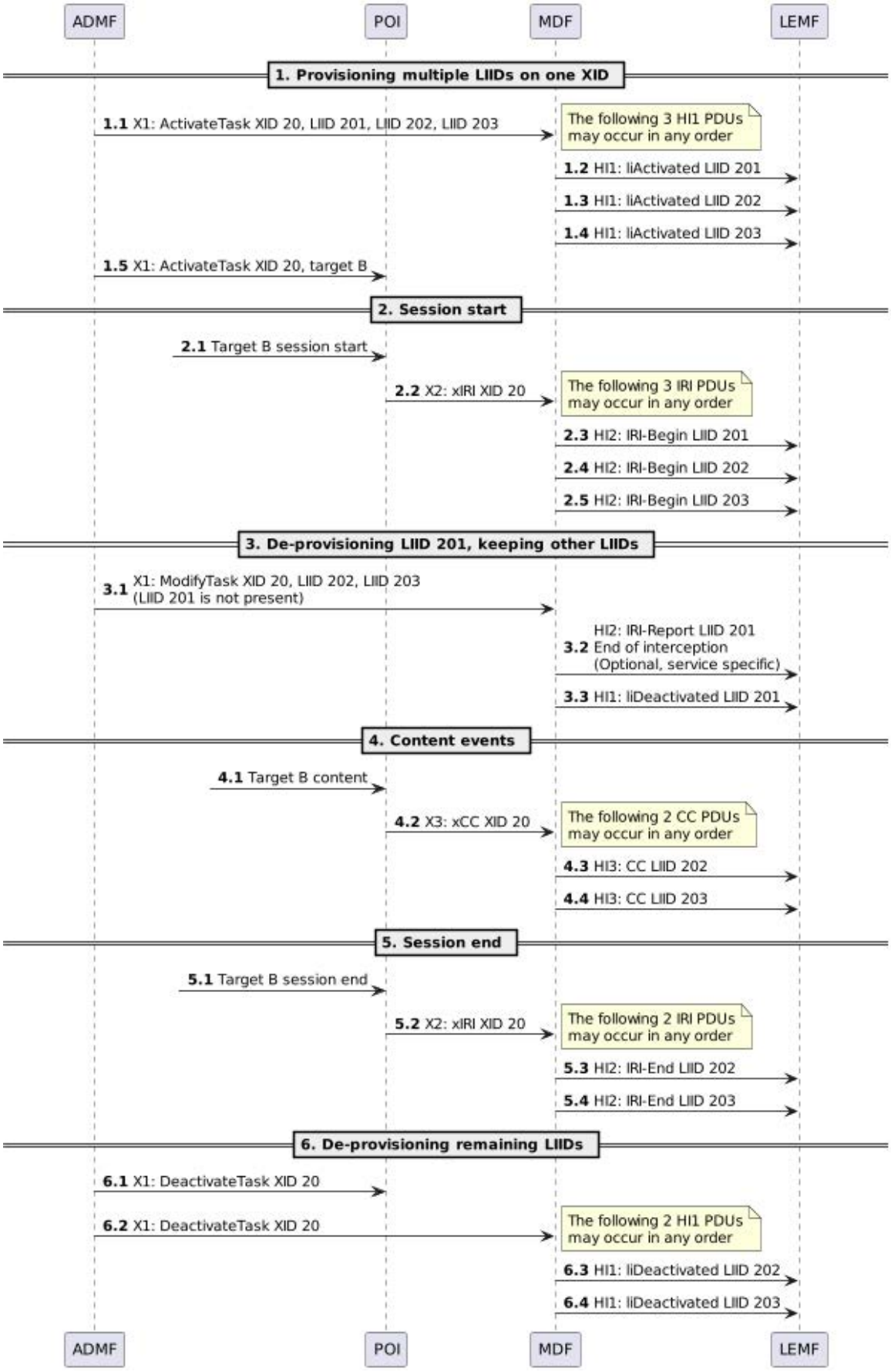

Figure I.2.2-1: Multiple LIIDs sharing an XID flows

The flow steps are:

1. ADMF provisions new Task for multiple LIIDs:

   1.1 ADMF sends MDF an X1 ActivateTask message for XID 20, LIID 201, LIID 202, and LIID 203. MDF is provisioned first to ensure that it is ready to receive X2/X3 PDUs.

   1.2 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 201.

   1.3 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 202.

   1.4 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 203.

   NOTE 1: PDUs in steps 1.2 to 1.4 may be generated and delivered in any order.

   1.5 ADMF sends POI an X1 ActivateTask message for XID 20, target B, without an LIID.

2. Session start event processed by POI:

   2.1 POI receives a session start event for Target B.

   2.2 POI sends MDF an xIRI PDU (over X2) for XID 20 containing the session start data.

   2.3 MDF sends LEMF an IRI-Begin PDU (over HI2) for LIID 201.

   2.4 MDF sends LEMF an IRI-Begin PDU (over HI2) for LIID 202.

   2.5 MDF sends LEMF an IRI-Begin PDU (over HI2) for LIID 203.

   NOTE 2: PDUs in steps 2.3 to 2.5 may be generated and delivered in any order.

   NOTE 3: The network session and content events may occur at any time during the lifetime of the Task.

3. ADMF deprovisions LIID 201 from the Task:

   3.1 ADMF sends MDF an X1 ModifyTask message for XID 20 only containing LIID 202 and LIID 203.

   3.2 MDF sends LEMF an IRI-Report PDU (over HI2) for LIID 201 end of interception, due to the active session. This is optional and service-specific.

   3.3 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 201.

4. Content events processed by POI:

   4.1 POI receives a content event for Target B.

   4.2 POI sends MDF an xCC PDU (over X3) for XID 20 containing the content.

   4.3 MDF sends LEMF a CC PDU (over HI3) for LIID 202.

   4.4 MDF sends LEMF a CC PDU (over HI3) for LIID 203.

   NOTE 4: PDUs in steps 4.3 and 4.4 may be generated and delivered in any order.

   NOTE 5: The network session and content events may occur at any time during the lifetime of the Task.

5. Session end event processed by POI:

   5.1 POI receives a session end event for Target B.

   5.2 POI sends MDF an xIRI PDU (over X2) for XID 20 containing the session end data.

   5.3 MDF sends LEMF an IRI-End PDU (over HI2) for LIID 202.

   5.4 MDF sends LEMF an IRI-End PDU (over HI2) for LIID 203.

   NOTE 6: The network session and content events may occur at any time during the lifetime of the Task.

   NOTE 7: PDUs in steps 5.3 and 5.4 may be generated and delivered in any order.

6. ADMF deprovisions Task:

   6.1 ADMF sends POI an X1 DeactivateTask message for XID 20. POI is deprovisioned first to ensure that no more X2/X3 PDUs are sent to MDF.

   6.2 ADMF sends MDF an X1 DeactivateTask message for XID 20.

   6.3 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 202.

   6.4 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 203.

NOTE 8: PDUs in steps 6.3 and 6.4 may be generated and delivered in any order.

I.2.3 Multiple LIIDs with separate XIDs

An example of tasking and operational flows for multiple LIIDs with separate XIDs per LIID for target C is shown in figure I.2.3-1. This shows the behaviour of the X3 AXRI conditional attribute (see ETSI TS 103 221-2 [19], clause 5.3.22).

NOTE 1: AXRI may be used only if explicitly permitted by the relevant LI architecture, and is an implementation decision in that case.

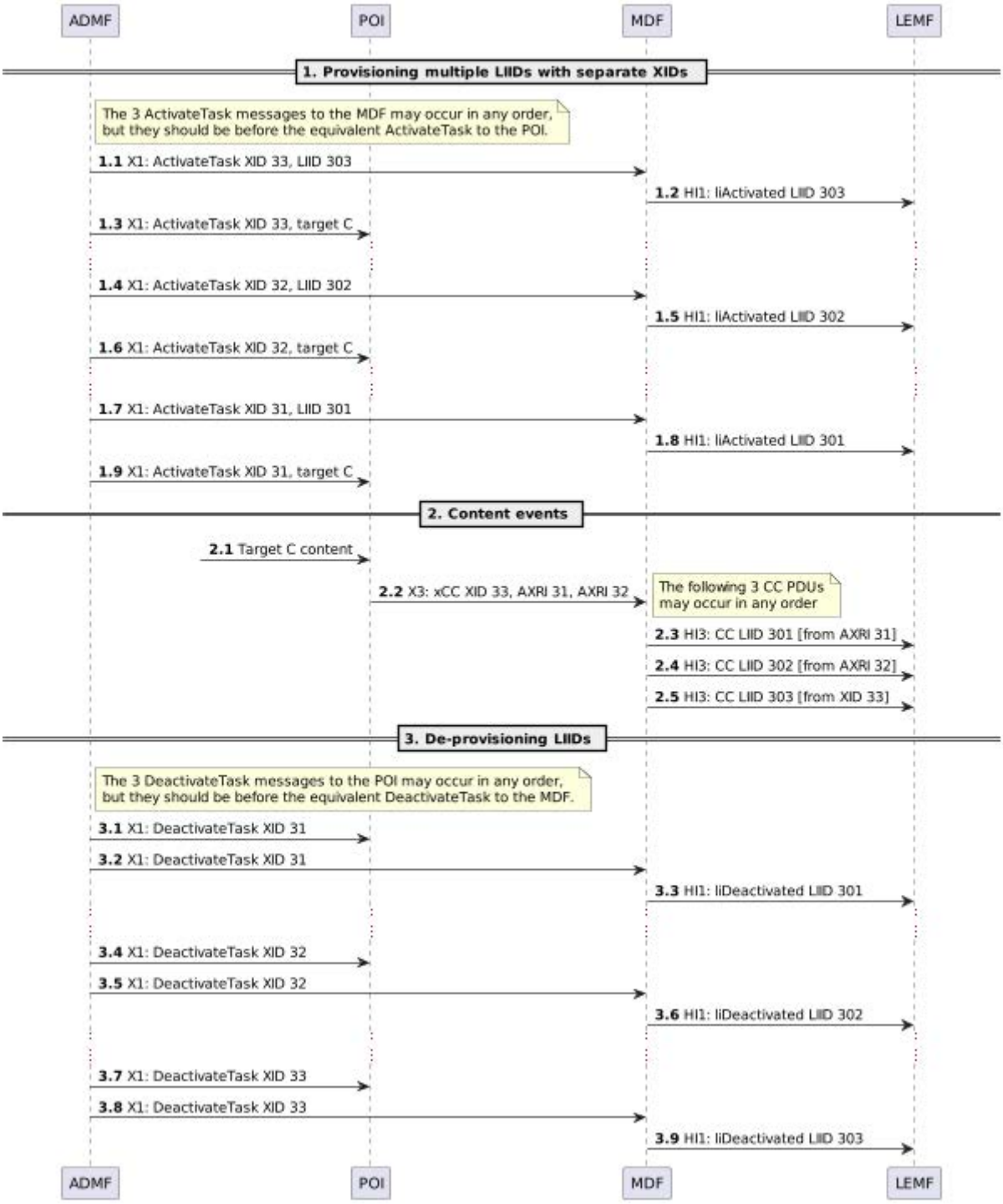

Figure I.2.3-1: Multiple LIIDs with separate XIDs flows

The flow steps are:

1. ADMF provisions new Task for each LIID:

   1.1 ADMF sends MDF an X1 ActivateTask message for XID 33, LIID 303. MDF is provisioned first to ensure that it is ready to receive X2/X3 PDUs.

   1.2 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 303.

   1.3 ADMF sends POI an X1 ActivateTask message for XID 33, target C, without an LIID.

   1.4 ADMF sends MDF an X1 ActivateTask message for XID 32, LIID 302.

1.5 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 302.

1.6 ADMF sends POI an X1 ActivateTask message for XID 32, target C, without an LIID.

1.7 ADMF sends MDF an X1 ActivateTask message for XID 31, LIID 301.

1.8 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 301.

1.9 ADMF sends POI an X1 ActivateTask message for XID 31, target C, without an LIID.

NOTE 2: The 3 ActivateTask messages (steps 1.1, 1.4 and 1.7) to MDF may occur in any order, but they should be before the equivalent ActivateTask messages to POI.

2. Network events processed by POI:

2.1 POI receives a content event for Target C.

2.2 POI sends MDF an xCC PDU (over X3) for XID 33, AXRI conditional attributes for XID 31 and XID 32, containing the content.

NOTE 3: The order of which XIDs are used for the xCC XID field versus the AXRI conditional attributes is implementation defined.

2.3 MDF sends LEMF a CC PDU (over HI3) for LIID 301 (matching AXRI for XID 31).

2.4 MDF sends LEMF a CC PDU (over HI3) for LIID 302 (matching AXRI for XID 32).

2.5 MDF sends LEMF a CC PDU (over HI3) for LIID 303 (matching XID 33).

NOTE 4: PDUs in steps 2.3 to 2.5 may be generated and delivered in any order.

NOTE 5: The network session and content events may occur at any time during the lifetime of the Task.

NOTE 6: An example flow for session events is not shown, but uses a similar pattern to content events.

3. ADMF deprovisions Tasks:

3.1 ADMF sends POI an X1 DeactivateTask message for XID 31. POI is deprovisioned first to ensure that no more X2/X3 PDUs are sent to MDF.

3.2 ADMF sends MDF an X1 DeactivateTask message for XID 31.

3.3 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 301.

3.4 ADMF sends POI an X1 DeactivateTask message for XID 32.

3.5 ADMF sends MDF an X1 DeactivateTask message for XID 32.

3.6 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 302.

3.7 ADMF sends POI an X1 DeactivateTask message for XID 33.

3.8 ADMF sends MDF an X1 DeactivateTask message for XID 33.

3.9 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 303.

NOTE 7: The 3 DeactivateTask messages (steps 3.1, 3.4 and 3.7) to POI may occur in any order, but they should be before the equivalent DeactivateTask to MDF.

I.2.4 Triggering and triggered POIs

An example of tasking and operational flows for a triggering deployment model (see clause 4.1.3) between a TF and a POI for target D is shown in figure I.2.4-1.

NOTE 1: This is an example scenario, and not prescriptive of triggering behaviour in a particular LI architecture.

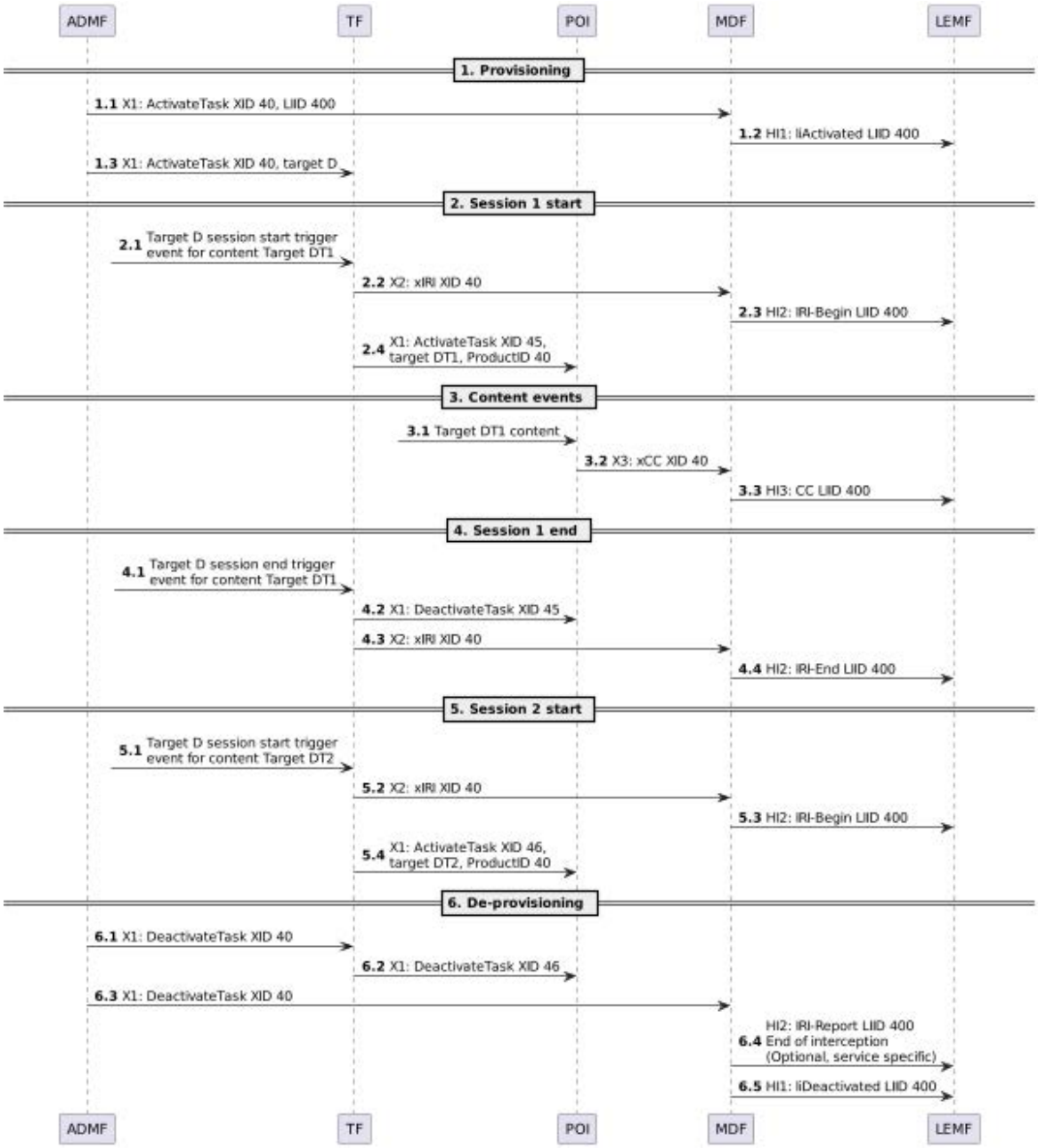

Figure I.2.4-1: Triggering and triggered flows

The flow steps are:

1 ADMF provisions new Task:

1.1 ADMF sends MDF an X1 ActivateTask message for XID 40, LIID 400. MDF is provisioned first to ensure that it is ready to receive X2/X3 PDUs.

1.2 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 400.

1.3 ADMF sends TF an X1 ActivateTask message for XID 40, target D, without an LIID.

2. Session 1 start event processed by TF:

2.1 TF receives a session start trigger event for target D, which maps to session-specific content target DT1.

2.2 TF (as POI) sends MDF an xIRI PDU (over X2) for XID 40 containing the session start data.

2.3 MDF sends LEMF an IRI-Begin PDU (over HI2) for LIID 400.

2.4 TF (as ADMF) sends POI an X1 ActivateTask message for XID 45, Target DT1, ProductID 40 (the original XID).

3. Content events processed by POI:

3.1 POI receives a content event for Target DT1.

3.2 POI sends MDF an xCC PDU (over X3) for XID 40 (the ProductID from XID 45) containing the content.

3.3 MDF sends LEMF a CC PDU (over HI3) for LIID 400.

NOTE 2: The network session and content events may occur at any time during the lifetime of the Task.

4. Session 1 end event processed by TF:

4.1 TF receives a session end trigger event for Target D.

4.2 TF (as ADMF) sends POI an X1 DeactivateTask message for XID 45.

4.3 TF (as POI) sends MDF an xIRI PDU (over X2) for XID 40 containing the session end data.

4.4 MDF sends LEMF an IRI-End PDU (over HI2) for LIID 400.

5. Session 2 start event processed by TF:

5.1 TF receives a session start trigger event for target D, which maps to session-specific content target DT2.

5.2 TF (as POI) sends MDF an xIRI PDU (over X2) for XID 40 containing the session start data.

5.3 MDF sends LEMF an IRI-Begin PDU (over HI2) for LIID 400.

5.4 TF (as ADMF) sends POI an X1 ActivateTask message for XID 46, Target DT2, ProductID 40 (the original XID).

6. ADMF deprovisions Task:

6.1 ADMF sends TF an X1 DeactivateTask message for XID 40. TF is deprovisioned first to ensure that no more X2/X3 PDUs are sent to MDF.

6.2 TF sends POI an X1 DeactivateTask message for XID 46.

6.3 ADMF sends MDF an X1 DeactivateTask message for XID 40.

6.4 MDF sends LEMF an IRI-Report PDU (over HI2) for LIID 400 end of interception, due to the active session. This is optional and service-specific.

6.5 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 400.

I.2.5 Triggering with multiple LIIDs and separate XIDs

An example of tasking and operational flows for a triggering deployment model (see clause 4.1.3) between a TF and a POI for target E, with multiple LIIDs with separate XIDs per LIID, is shown in figure I.2.5-1.

NOTE 1: This is an example scenario, and not prescriptive of triggering behaviour in a particular LI architecture.

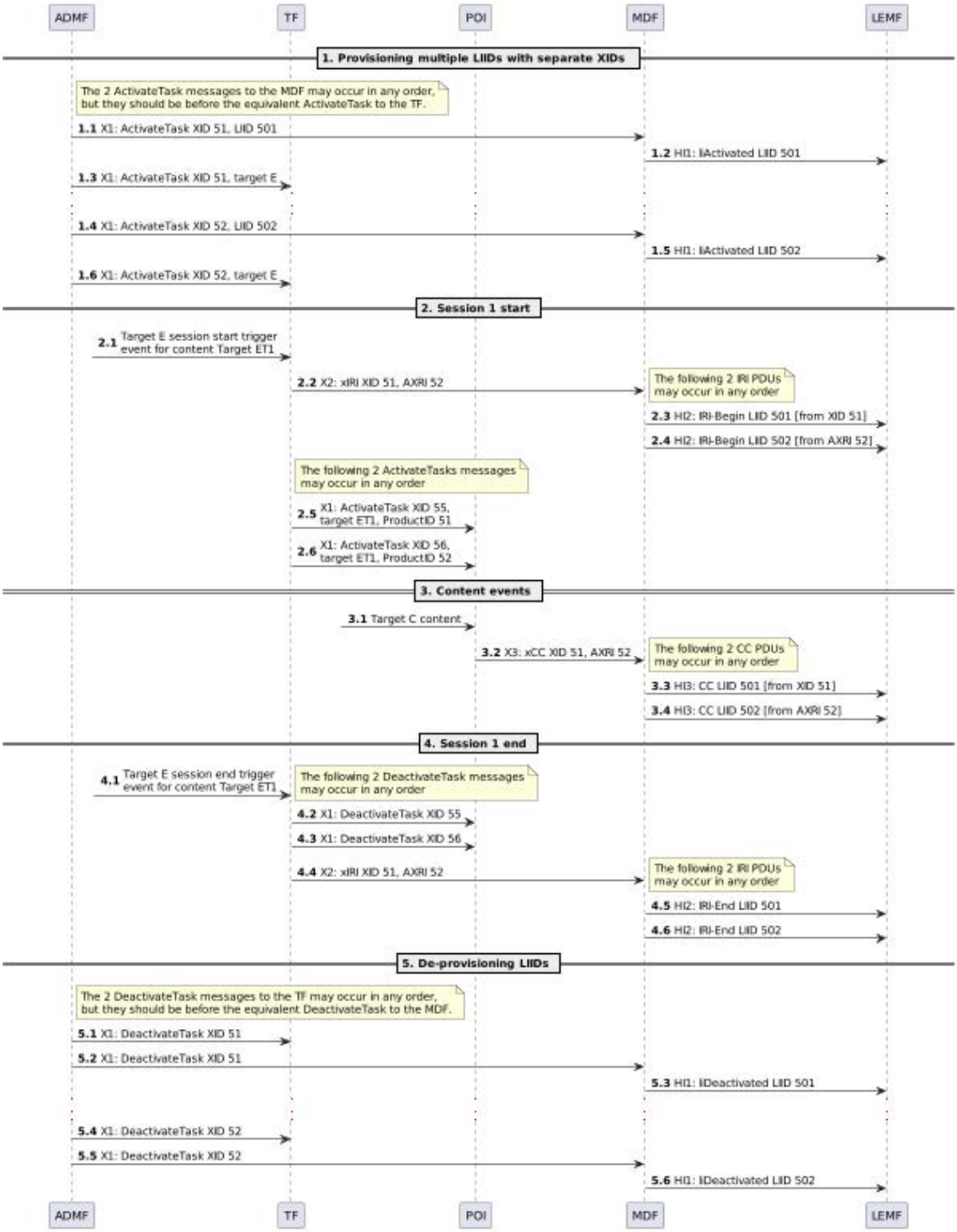

Figure I.2.5-1: Triggering with multiple LIIDs and separate XIDs

The flow steps are:

1. ADMF provisions new Task for each LIID:

   1.1 ADMF sends MDF an X1 ActivateTask message for XID 51, LIID 501. MDF is provisioned first to ensure that it is ready to receive X2/X3 PDUs.

   1.2 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 501.

   1.3 ADMF sends POI an X1 ActivateTask message for XID 51, target E, without an LIID.

   1.4 ADMF sends MDF an X1 ActivateTask message for XID 52, LIID 502.

1.5 MDF sends LEMF an HI1-Notification liActivated PDU (over HI1) for LIID 502.

1.6 ADMF sends POI an X1 ActivateTask message for XID 52, target E, without an LIID.

NOTE 2: The 2 ActivateTask messages (steps 1.1 and 1.4) to MDF may occur in any order, but they should be before the equivalent ActivateTask messages to POI.

2. Session 1 start event processed by TF:

2.1 TF receives a session start trigger event for target E, which maps to session-specific content target ET1.

2.2 TF (as POI) sends MDF an xIRI PDU (over X2) for XID 51, AXRI conditional attribute for XID 52, containing the session start data.

NOTE 3: The order of which XIDs are used for the xIRI XID field versus the AXRI conditional attributes is implementation defined.

2.3 MDF sends LEMF an IRI-Begin PDU (over HI2) for LIID 501 (for XID 51).

2.4 MDF sends LEMF an IRI-Begin PDU (over HI2) for LIID 502 (matching AXRI for XID 52).

NOTE 4: PDUs in steps 2.3 and 2.4 may be generated and delivered in any order.

2.5 TF (as ADMF) sends POI an X1 ActivateTask message for XID 55, Target ET1, ProductID 51 (the original XID for MDF LIID 501).

2.6 TF (as ADMF) sends POI an X1 ActivateTask message for XID 56, Target ET1, ProductID 52 (the original XID for MDF LIID 502).

NOTE 5: The 2 ActivateTask messages (steps 2.5 and 2.6) to POI may occur in any order.

3. Content events processed by POI:

3.1 POI receives a content event for Target ET1.

3.2 POI sends MDF an xCC PDU (over X3) for XID 51 (the ProductID from XID 55), AXRI conditional attribute for XID 52 (the ProductID from XID 56), containing the content.

NOTE 6: The order of which XIDs are used for the xCC XID field versus the AXRI conditional attributes is implementation defined.

3.3 MDF sends LEMF a CC PDU (over HI3) for LIID 501 (for XID 51).

3.4 MDF sends LEMF a CC PDU (over HI3) for LIID 502 (matching AXRI for XID 52).

NOTE 7: PDUs in steps 3.3 and 3.4 may be generated and delivered in any order.

NOTE 8: The network session and content events may occur at any time during the lifetime of the Task.

4. Session 1 end event processed by TF:

4.1 TF receives a session end trigger event for Target E.

4.2 TF (as ADMF) sends POI an X1 DeactivateTask message for XID 55.

4.3 TF (as ADMF) sends POI an X1 DeactivateTask message for XID 56.

NOTE 9: The 2 DeactivateTask messages (steps 4.2 and 4.3) to POI may occur in any order.

4.4 TF (as POI) sends MDF an xIRI PDU (over X2) for XID 51, AXRI conditional attribute for XID 52, containing the session end data.

NOTE 10: The order of which XIDs are used for the xIRI XID field versus the AXRI conditional attributes is implementation defined.

4.5 MDF sends LEMF an IRI-End PDU (over HI2) for LIID 501 (for XID 51).

4.6 MDF sends LEMF an IRI-End PDU (over HI2) for LIID 502 (for XID 52).

NOTE 11: PDUs in steps 4.5 and 4.6 may be generated and delivered in any order.

5. ADMF deprovisions Tasks:

   5.1 ADMF sends POI an X1 DeactivateTask message for XID 51. POI is deprovisioned first to ensure that no more X2/X3 PDUs are sent to MDF.

   5.2 ADMF sends MDF an X1 DeactivateTask message for XID 51.

   5.3 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 501.

   5.4 ADMF sends POI an X1 DeactivateTask message for XID 52.

   5.5 ADMF sends MDF an X1 DeactivateTask message for XID 52.

   5.6 MDF sends LEMF an HI1-Notification liDeactivated PDU (over HI1) for LIID 502.

NOTE 12: The 2 DeactivateTask messages (steps 5.1 and 5.4) to POI may occur in any order, but they should be before the equivalent DeactivateTask to MDF.

Annex J (informative): Change history

Status of the present document: ETSI TS 103 221-1 Internal Network Interfaces; Part 1: X1		
TC LI Approval Date	**Version**	**Remarks**
October 2017	1.1.1	First publication
February 2018	1.2.1	Included Change Request: TS103221-1CR001r1 (cat F) Warning and Faults Reporting This CR was approved by TC LI#47 (5-7 February 2018, New Delhi)
June 2018	1.3.1	Included Change Request: TS103221-1CR002r2 (cat F) X1 response/request lifecycle This CR was approved by TC LI#48 (26-28 June 2018, Bergen)
February 2019	1.4.1	Included Change Request: TS103221-1CR003r3 (cat B) Support for 5G This CR was approved by TC LI#50 (5-7 February 2019, Dubai)
July 2019	1.5.1	Included Change Requests: CR004r1 (cat F) Permitting multiple extensions in X1 CR005r6 (cat C) Mediation Details Update CR006r1 (cat F) Task Details Update CR007r1 (cat F) Clarify XID to LIID Relationship CR008r1 (cat F) DeliveryAddress Updates CR009r1 (cat F) TaskStatus Updates CR010 (cat C) Corrections after implementation These CRs were approved by TC LI#51 (11-13 June 2019, Texel)
October 2019	1.6.1	Included Change Requests: CR012 (cat B) Use of HTTP/2 CR013 (cat B) Addition of Product ID CR014 (cat C) Making the requirements annex informative CR015 (cat B) Update for TLS 1.3 CR016 (cat D) Alignment to 3GPP terminology These CRs were approved by TC LI#52 (15-17 October 2019, Turin)
July 2020	1.7.1	Included Change Request: CR017 (cat F) Clarifications on use of delayed Acknowledgements for Destinations This CR was approved by TC LI#54-e (17-25 June 2020)
February 2021	1.8.1	Included Change Requests: CR019r2 (cat F) MessageTimestamp clarification CR020r2 (cat F) X1 HTTP path clarification CR021r2 (cat F) Updating the version field CR022r3 (cat F) Clarifying UID RDN These CRs were approved by TC LI#56-e (15-19 February 2021)
June 2021	1.9.1	Included Change Requests: CR025r1 (cat B) Addition of InternationalizedEmailAddress type CR026r1 (cat B) New services in service scope structure of table C.2.2-2 These CRs were approved by TC LI#57-e (21-25 June 2021)
October 2021	1.10.1	Included Change Requests: CR027r2 (cat B) Generic object mechanism CR028r2 (cat B) Hashed Identifiers CR029r2 (cat B) Addition of EUI-64 CR030r2 (cat B) Addition of Service Type to Task Details These CRs were approved by TC LI#58-e (18-22 October 2021)

Status of the present document: ETSI TS 103 221-1 Internal Network Interfaces; Part 1: X1		
TC LI Approval Date	**Version**	**Remarks**
February 2022	1.11.1	Included Change Requests: CR031r2 (cat B) Clarifications to Generic Object sections CR032r3 (cat C) Destination Identifier Set Object These CRs were approved by TC LI#59-e (14-19 February 2022)
July 2022	1.12.1	Included Change Requests: CR034r4 (cat C) Adding NE alert when database is cleared CR035r1 (cat C) ErrorResponse for X1 RequestMessageType extensions These CRs were approved by TC LI#60 (28-30 June 2022, Paris)
November 2022	1.13.1	Included Change Requests: CR036r2 (cat C) Enhancing getting details from NE CR037r2 (cat B) ServiceScopingOptions Alignment These CRs were approved by TC LI#61 (22-24 September 2022, Malmö)
March 2023	1.14.1	Included Change Request: CR039 (cat B) Extension point for TaskStatus This CR was approved by TC LI#62 (31 January – 2 February 2023, Sophia Antipolis)
June 2023	1.15.1	Included Change Requests: CR040r1 (cat C) Additional options for Keepalive behaviour CR041r3 (cat B) Traffic Policies in X1 CR042r1 (cat C) Request Message Type Identification CR043r1 (cat D) Clarifications for DeactivateAllTasks CR050r1 (cat C) Stricter XSD requirements These CRs were approved by TC LI#63 (20-22 June 2023, Rome)
December 2023	1.16.1	Included Change Requests: CR052r4 (cat C) New TaskReportType for remote POI reporting CR053r3 (cat C) Clarification of EndTime Procedures CR054r4 (cat C) Loss of X1 Actions CR055r1 (cat C) Extensions for ReportTaskIssue and ReportNEIssue CR057r4 (cat F) Differentiating NE from NF These CRs were approved by TC LI#64 (31 October – 2 November 2023, Sydney)
February 2024	1.17.1	Included Change Requests: CR058r6 (cat F) List Of Associated DIDs Correction CR059r6 (cat C) Generic Target Identifier CR060 (cat F) Samples only in Forge These CRs were approved by TC LI#65 (6-8 February 2024, Saariselkä)
June 2024	1.18.1	Included Change Requests: CR062r2 (cat B) X1 Certificate Binding CR063 (cat F) Introducing X1 Context concept These CRs were approved by TC LI#66 (18-21 June 2024, Lucerne)
October 2024	1.19.1	Included Change Request: CR064r5 (cat B) Defining X1 Configuration parameters This CR was approved by TC LI#67 (22-24 October 2024, Vancouver)
February 2025	1.20.1	Included Change Requests: CR066r1 (cat C) Updated Format Descriptions for ETSI Target Identifiers CR067r2 (cat F) Corrections on Keepalive functionality CR068 (cat F) Correction on ReportNEIssueRequest parameter IssueCode CR069r2 (cat C) Destination Set Delivery Type These CRs were approved by TC LI#68 (25-27 February 2025, Dublin)
June 2025	1.21.1	Included Change Request: CR071r2 (cat B) Tasking flows annex This CR was approved by TC LI#69 (3-5 June 2025, Trondheim) Numbering of tables and figures adjusted.

Status of the present document: ETSI TS 103 221-1 Internal Network Interfaces; Part 1: X1		
TC LI Approval Date	**Version**	**Remarks**
October 2025	1.22.1	Included Change Requests: CR072r3 (cat B) Update to add TCPPortList and UDPPortList to Target Identifiers CR073r1 (cat F) Corrections on Destinations and Destination Sets CR074r1 (cat B) Addition of VRF to TargetIdentifiers These CRs were approved by TC LI#70 (30 September - 2 October 2025, New York)
January 2026	1.23.1	Included Change Requests: CR077r2 (cat F) Improvement of References in Table 6.2.1.2-2 TargetIdentifier Formats CR078r1 (cat B) Selective IRI Delivery Provisioning These CRs were approved by TC LI#71 (21-23 January 2026, Sophia Antipolis)

History

Version	Date	Status
V1.1.1	October 2017	Publication
V1.2.1	March 2018	Publication
V1.3.1	September 2018	Publication
V1.4.1	April 2019	Publication
V1.5.1	July 2019	Publication
V1.6.1	December 2019	Publication
V1.7.1	August 2020	Publication
V1.8.1	April 2021	Publication
V1.9.1	July 2021	Publication
V1.10.1	December 2021	Publication
V1.11.1	March 2022	Publication
V1.12.1	August 2022	Publication
V1.13.1	December 2022	Publication
V1.14.1	March 2023	Publication
V1.15.1	August 2023	Publication
V1.16.1	January 2024	Publication
V1.17.1	April 2024	Publication
V1.18.1	July 2024	Publication
V1.19.1	December 2024	Publication
V1.20.1	May 2025	Publication
V1.21.1	August 2025	Publication
V1.22.1	November 2025	Publication
V1.23.1	March 2026	Publication