ETSI TS 103 836-4-1 V2.1.1 (2022-11)

**Intelligent Transport Systems (ITS);
Vehicular Communications;
GeoNetworking;
Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications;
Sub-part 1: Media-Independent Functionality;
Release 2**

Reference
DTS/ITS-003203

Keywords
architecture, ITS

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 Web server (https://ipr.etsi.org/).

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™** and **LTE™** 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 Intelligent Transport Systems (ITS).

The present document is part 4, sub-part 1 of a multi-part deliverable covering the Vehicular Communications; GeoNetworking, as identified below:

Part 1: "Requirements";

Part 2: "Scenarios";

Part 3: "Network Architecture";

Part 4: "Geographical addressing and forwarding for point-to-point and point-to-multipoint communications";

Sub-part 1: "Media-Independent Functionality; Release 2";

Sub-part 2: "Media-dependent functionalities for ITS-G5";

Sub-part 3: "Media-dependent functionalities for NR-V2X PC5 and LTE-V2X PC5; Release 2";

Part 5: "Transport Protocols";

Part 6: "Internet Integration".

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.

Introduction

The GeoNetworking protocol is a network layer protocol that provides packet routing in an ad hoc network. It makes use of geographical positions for packet transport. GeoNetworking supports the communication among individual ITS stations as well as the distribution of packets in geographical areas.

GeoNetworking can be executed over different ITS access technologies for short-range wireless technologies, such as ITS-G5 and infrared. The ITS access technologies for short-range wireless technologies have many technical commonalities, but also differences. In order to reuse the GeoNetworking protocol specification for multiple ITS access technologies, the specification is separated into media-independent and media-dependent functionalities. Media-independent functionalities are those which are common to all ITS access technologies for short-range wireless communication to be used for GeoNetworking. The media-dependent functionalities extend the media-independent functionality for a specific ITS access technology. Therefore, the GeoNetworking protocol specification consists of the standard for media-independent functionality and at least one standard for media-dependent functionality. However, it should be noted that the media-dependent extensions do not represent distinct protocol entities.

1 Scope

The present document specifies the Release 2 of the GeoNetworking protocol. Compared to the Release 1 specification, Release 2 primarily adds support for multi-channel operation.

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 at https://docbox.etsi.org/Reference/.

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 103 899: "Intelligent Transport Systems (ITS); Vehicular Communications; Geographical Area Definition; Release 2".

[2] ETSI TS 102 894-2: "Intelligent Transport Systems (ITS); Users and applications requirements; Part 2: Applications and facilities layer common data dictionary; Release 2".

[3] ETSI TS 103 836-5-1: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 5: Transport Protocols; Sub-part 1: Basic Transport Protocol; Release 2".

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 are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 103 836-3: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 3: Network architecture; Release 2".

[i.2] ETSI TS 103 697 (V2.1.1): "Intelligent Transport Systems (ITS); Architecture; Multi-Channel Operation (MCO) for Cooperative ITS (C-ITS); Release 2".

[i.3] National Imagery and Mapping Agency (NIMA), US Department of Defense: "World Geodetic System 1984 - Its Definition and Relation with Local Geodetic Systems", Third Edition - Amendment 1, NIMA TR 8350.2.

[i.4] Void.

[i.5] IEEE 802.3:2008™: "IEEE Standard for Information Technology - Telecommunications and information exchange between systems-Local and metropolitan area networks - Specific requirements - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications".

[i.6] ETSI TS 102 965: "Intelligent Transport Systems (ITS); Application Object Identifier (ITS-AID); Registration; Release 2".

[i.7] ETSI TS 103 695: "Intelligent Transport Systems (ITS); Access layer specification in the 5 GHz frequency band; Multi-Channel Operation (MCO) for Cooperative ITS (C-ITS); Release 2".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI TS 103 836-3 [i.1], ETSI TS 103 697 [i.2] and the following apply:

destination: receiver that processes a packet and delivers it to upper protocol entities, but does not relay the packet to other GeoAdhoc routers

forwarder: GeoAdhoc router that processes a packet and relays it to other GeoAdhoc routers

GeoAdhoc router: networking functionality that implements the GeoNetworking protocol

local position vector: position vector for the local GeoAdhoc router

media dependent procedures: packet handling steps that are specific to the Access Layer technology over which the GeoNetworking packet will be transmitted

neighbour: GeoAdhoc router in direct (single-hop) communication range

packet: GeoNetworking PDU

packet transport type: method of handling GeoNetworking packets

position accuracy indicator: binary that indicates whether a position is within a specific confidence interval

position vector: position information of a GeoAdhoc router represented by a tuple of address, timestamp, geographical position, speed, heading and corresponding accuracy information

receiver: GeoAdhoc router that processes a packet, delivers its data to upper protocol entities

sender: GeoAdhoc router that has sent the GeoNetworking packet

source: GeoAdhoc router that originates a GeoNetworking packet

traffic class: identifier assigned to a GeoNetworking packet that expresses its requirements on data transport

3.2 Symbols

For the purposes of the present document, the symbols given in ETSI TS 103 697 [i.2] and the following apply:

ALI_ADDR	Address of the Access Layer instance that identifies the ITS-S at the link layer protocol entity in the Access Layer
GEO_MAX	Maximum size of the GeoNetworking packet header
H(GN_ADDR)	Heading of the ITS-S GN_ADDR
LAT	Latitude
LL_ADDR_NH	Address of the link layer of of the next hop
LONG	Longitude
LS_PENDING	Location Service pending flag
MTU_AL	MTU of the Access Layer
PAI(POS, GN_ADDR)	Position accuracy indicator for geographical position POS of the ITS-S GN_ADDR
PDR(GN_ADDR)	Packet data rate (exponential moving average)
POS(GN_ADDR)	Geographical position of the ITS-S GN_ADDR
PV(GN_ADDR)	Position vector of the ITS-S GN_ADDR

RAND[x,y]	Function that returns a random (integer) number from a uniform distribution in the given interval [x,y]
S(GN_ADDR)	Speed of the ITS-S GN_ADDR
SN_MAX	Largest possible value of the sequence number
SN(P)	Value of the sequence number field carried in a GeoNetworking packet
T(LocTE)	Lifetime of an entry in the location table
TO_CBF_MIN	Timeout; minimum duration a packet is buffered in the CBF cache
TO_CBF_MAX	Timeout; maximum duration a packet is buffered in the CBF cache
TST(GN_ADDR)	Last timestamp received from a GeoAdhoc router
TST(P)	Value of the timestamp field carried in a GeoNetworking packet
TST(TAI)	Number of elapsed TAI milliseconds since 2004-01-01 00:00:00.000 UTC

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI TS 103 836-3 [i.1], ETSI TS 103 697 [i.2] and the following apply:

ALI	Access Layer Instance
AliID	Access layer instance Identifier
ARH	ALI Request Handler
ASN	Abstract Syntax Notation
BC	BroadCast
BTP	Basic Transport Protocol
CBF	Contention-Based Forwarding
CLR	Channel Load Ratio
DAD	Duplicate Address Detection
DE	Destination
DPC	Duplicate Packet Counter
DPD	Duplicate Packet Detection
DPL	Duplicate Packet List
EMA	Exponential Moving Average
EPV	Ego Position Vector
FIFO	First In First Out
GAC	Geographically-Scoped Anycast
GAGH	GeoNetworking Access Layer Instance Group Handler
GBC	Geographically-Scoped Broadcast
GF	Greedy Forwarding
GN	GeoNetworking
GN_ADDR	GeoNetworking ADDRess
GN-PDU	GeoNetworking Protocol Data Unit
GN-SDU	GeoNetworking Service Data Unit
GN6ASL	GeoNetworking to IPv6 Adaptation Sub-Layer
GN6-SDU	GN6 Service Data Unit
GUC	Geographically-Scoped Unicast
HST	Header Sub-Type
HT	Header Type
LocT	Location Table
LocTE	Location Table Entry
LPV	Local Position Vector
LS	Location Service
LT	LifeTime
LTE	Long Term Evolution
MAC	Medium Access Control
MFR	Most Forward within Radius
MHL	Maximum Hop Limit
MHVB	Multi-Hop Vehicular Broadcast
MIB	Management Information Base
MID	MAC ID
MPU	MCO Parameters Update
MTU	Maximum Transmit Unit
NH	Next Header

NU	Network Updates
PAI	Position Accuracy Indicator
PDR	Packet Data Rate
PDU	Protocol Data Unit
PL	Payload Length
POS	POSition
PV	Position Vector
RHL	Remaining Hop Limit
RTC	Retransmit Counter
SCF	Store Carry & Forward
SDU	Service Data Unit
SE	SEnder
SHB	Single Hop Broadcast
SN	Sequence Number
SO	SOurce
SPV	Short Position Vector
SSP	Service Specific Permissions
TAI	Temps Atomique International (International Atomic Time)
TC	Traffic Class
TC ID	Traffic Class Identifier
TPT	Traffic Participant Type
TSB	Topologically Scoped Broadcast
T-SDU	Transport Service Data Unit
TST	TimeSTamp
UC	UniCast
UTC	Universal Time Coordinated
V2X	Vehicle-to-Everything
WGS	World Geodetic System

4 Services provided by the GeoNetworking protocol

The GeoNetworking protocol is a network protocol that resides in the Networking & Transport layer. Requirements on the GeoNetworking protocol are specified in ETSI TS 103 836-3 [i.1]. It is executed in the ad hoc router, specifically in the GeoAdhoc router. It provides the transport of packets in the ITS ad hoc network (ETSI TS 103 836-3 [i.1]).

GeoNetworking comprises several components, namely GeoNetworking Core (GN Core), GeoNetworking ALI Group Handlers (GAGHs) and MCO_NET (figure 1). GN Core provides channel- and media-independent GeoNetworking functionalities. A GAGH realizes channel dependent, media-dependent, and media-independent GeoNetworking functionalities. The MCO_NET entity is optional and provides MCO management functionalities as part of the GeoNetworking protocol for ITS-Ss that support multiple communication channels on one or more access layer technologies.

The GeoNetworking protocol provides services to upper protocol entities, i.e. the transport protocol, such as the Basic Transport Protocol (BTP), the GeoNetworking to IPv6 Adaptation Sub-Layer (GN6ASL), and MCO-related services to the Facilities layer entities.

In order to provide its packet transport services, the GeoNetworking protocol uses the services of the Access Layer via the NET_ACC_DATA interface. To provide MCO functionalities to upper layers, MCO_NET uses services provided by the Access layer via the NET_ACC_MMT interface.

NOTE: When data are delivered via the NET_ACC_DATA interface and uses MCO, the interface may be also called MCO_NET_ACC_DATA (see e.g. ETSI TS 103 697 [i.2]).

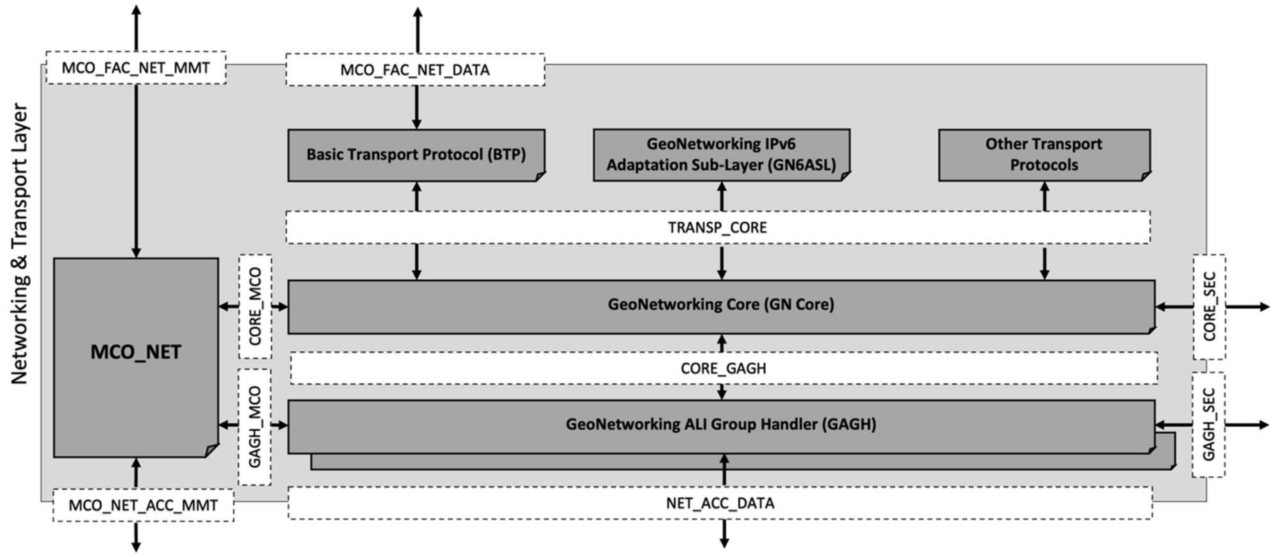

Figure 1: GeoNetworking-related components and interfaces

The present document specifies the internal interfaces of the Networking & Transport layer as defined in ETSI TS 103 836-3 [i.1] (figure 1), i.e.:

- the interface between the GeoNetworking protocol and the ITS transport protocol, such as the Basic Transport Protocol (BTP), the GeoNetworking IPv6 Adaptation Sub-Layer (GN6ASL), and other transport protocols;
- the interface between the GeoNetworking Core and the GeoNetworking ALI Group Handler;
- the interface between the GeoNetworking Core and the MCO_NET entity;
- the interface between the GeoNetworking ALI Group Handlers and the MCO_NET entity.

To provide MCO services to the Facilities layer entities, the present document specifies the MCO_FAC_NET_MMT interface between the Networking & Transport layer and the Facilities layer.

5 Format convention

The basic convention for the specification of packet formats is illustrated in figure 2. The bits are grouped into octets. The bits of an octet are always shown horizontally and are numbered from 0 to 7. Up to 4 octets are shown horizontally; multiple sets of 4 octets are grouped vertically. Octets are numbered from 0 to N-1.

<table>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">Octet 0</td><td colspan="8">Octet 1</td><td colspan="8">Octet 2</td><td colspan="8">Octet 3</td></tr>
<tr><td colspan="32">Octet 4 to Octet 7</td></tr>
<tr><td colspan="24">…</td><td colspan="8">Octet N-1</td></tr>
</table>

Figure 2: Format convention

When (a part of) an Octet represents a numeric quantity the left most bit in the diagram is the most significant bit (Big Endian). Similarly, when a numeric value spans multiple octet fields the left most field is the most significant.

Octets are transmitted in ascending numerical order (left to right).

EXAMPLE: The decimal value 199 shall be represented as shown below.

0	1	2	3	4	5	6	7
1	1	0	0	0	1	1	1

6 GeoNetworking address

6.1 General

Every GeoAdhoc router, irrespective of the number of GAGHs, shall have a unique GeoNetworking address. This address shall be used in the header of a GeoNetworking packet and identify the communicating GeoNetworking entities. In order to ensure the uniqueness of the GeoNetworking address, duplicate detection as specified in clause 10.2.1.5 is applied. For multi-channel operation, an ITS-S with several GAGHs shall use a single GeoNetworking address.

NOTE: In case the GN MID changes, the link-layer address should also be changed. In one possible implementation the link-layer address is set from the GeoNetworking protocol entity.

6.2 GeoNetworking address format

The format of the GeoNetworking address shall be as described in figure 3.

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td>M</td><td colspan="5">ST</td><td colspan="10">Reserved</td><td colspan="16">MID</td></tr>
<tr><td colspan="32">MID</td></tr>
</table>

Figure 3: GeoNetworking address format

6.3 Fields of the GeoNetworking address

The GeoNetworking address shall be comprised of the fields specified in table 1.

Table 1: Fields of the GeoNetworking address

Field #	Field name	Octet/bit position		Type	Description
		First	Last		
1	M	Octet 0 Bit 0	Octet 0 Bit 0	1 bit unsigned integer	This bit allows distinguishing between manually configured network address (clause 10.2.1.3.3) (update) and the initial GeoNetworking address (clause 10.2.1.3.2). M is set to 1 if the address is manually configured otherwise it equals 0.
2	TPT	Octet 0 Bit 1	Octet 0 Bit 5	5 bit unsigned integer	Traffic participant type To identify the traffic participant type: 0 - unknown 1 - pedestrian 2 - cyclist 3 - moped 4 - motorcycle 5 - passengerCar 6 - bus 7 - lightTruck 8 - heavy Truck 9 - trailer 10 - specialVehicle 11 - tram 12 - lightVruVehicle 13 - animal 14 - agricultural 15 - roadSideUnit (see note).
3	Reserved	Octet 0 Bit 6	Octet 1 Bit 7	10 bit unsigned integer	Reserved
4	MID	Octet 2	Octet 7	48 bit address	Used to derive the Link Layer address.
NOTE: The values of the TPT are aligned with ETSI TS 102 894-2 [2].					

The first bit is reserved for the recognition of manual configured GeoNetworking addresses.

The MID field shall be used to derive an Access layer address.

In order to allow for the resolution of a GeoUnicast destination GN_ADDR from an IPv6 destination address using virtual interfaces of type Ethernet V2.0/IEEE 802.3 LAN [i.5], the GeoNetworking address space shall remain 48-bit wide (size of the MID field in the GeoNetworking address). In particular, the GN6ASL resolves an MID from a unicast destination IPv6 address and passes it to GeoNetworking via the service primitive *TRANSP_CORE.request* (clause J.2). Then, the GeoNetworking protocol is responsible for deriving a full GN_ADDR from the MID. This full GN_ADDR shall be derived from a LocTE (if it exists) or by executing the Location Service with Request GN_ADDR field containing only the MID part and the other bits set to 0.

To be compliant with the IPv6 over GeoNetworking architecture, the GeoNetworking address space shall remain 48-bit wide (size of the MID field in the GeoNetworking address) in order to provide a virtual interface of Ethernet type to IPv6 and to perform the forwarding via GeoNetworking in a transparent way.

If the address is updated for privacy reasons, i.e. by assignment of an alias identity, only the last field of the address shall be updated and derived from the alias identity.

7 Security and privacy

The GeoNetworking protocol may exchange information with the security entity via the CORE_SEC and GAGH_SEC (see figure 1) interfaces when required.

8 Data structures

8.1 General

A GeoAdhoc router shall maintain different types of local data structures per GAGH, i.e.:

- location table (clause 8.2);
- sequence number (clause 8.4);
- location service packet buffer (clause 8.5);
- forwarding packet buffer (clause 8.6).

A GeoAdhoc router shall also maintain a data structure for the ego position vector (clause 8.3) independently of the GAGH.

NOTE: Compared to the other data structures, it is assumed that the ego position vector is independent of GAGH and therefore expected to be implemented in the GN Core (see figure 1).

8.2 Location table

8.2.1 General

A GeoAdhoc router shall maintain a local data structure, referred to as Location Table (LocT). This data structure holds information about other ITS-Ss received by the same GAGH that execute the GeoNetworking protocol. The data elements of a location table entry are specified in clause 8.2.2 and the maintenance of the location table in clause 8.2.3.

8.2.2 Minimum data elements of a *Location Table Entry*

A *Location Table Entry* (LocTE) shall contain at least the following data elements:

- GeoNetwork address of the ITS-S *GN_ADDR*.
- Link layer address of the ITS-S.
- Type of the ITS-S (e.g. vehicle ITS-S, roadside ITS-S).
- Version of the GeoNetworking protocol used by the ITS-S.
- Position vector *PV*, i.e. *Long Position Vector LPV* (clause 9.5.2), of the ITS-S, comprised of:
    - Geographical position *POS(GN_ADDR)*;
    - Speed *S(GN_ADDR)*;
    - Heading *H(GN_ADDR)*;
    - Timestamp of the geographical position *TST(POS, GN_ADDR)*;
    - Position accuracy indicator *PAI(POS, GN_ADDR).*
- Flag *LS_PENDING(GN_ADDR)*:
    - Flag indicating that a Location Service (LS) (clause 10.2.4) is in progress.
- Flag *IS_NEIGHBOUR(GN_ADDR)*:
    - Flag indicating that the GeoAdhoc router is in direct communication range, i.e. is a neighbour.

- *DPL(GN_ADDR)*:
  - Duplicate packet list for source GN_ADDR.
- Timestamp *TST(GN_ADDR*):
  - The timestamp of the last packet from the source *GN_ADDR* that was identified as 'not duplicated'.
- Packet data rate *PDR(GN_ADDR)* as Exponential Moving Average (EMA) (clause B.2).

NOTE 1: The LocTE may contain more data elements defined in media-dependent functionalities of GeoNetworking.

NOTE 2: The format of the data in the LocT is implementation-specific and, therefore, not further specified.

NOTE 3: *LS_PENDING(GN_ADDR*) equals TRUE indicates that for the *GN_ADDR* a location service has been invoked and is in process.

8.2.3 Maintenance of the Location Table

The entries in the location table shall be soft-state, i.e. entries are added with a lifetime *T(LocTE)* set to the value of the GN protocol constant `itsGnLifetimeLocTE` and shall be removed when the lifetimes expire.

The flag *LS_PENDING(GN_ADDR)* shall be soft-state, i.e. it shall be unset when the flag is not renewed within the lifetime 3 × `itsGnBeaconServiceRetransmitTimer`.

8.3 Ego Position Vector

8.3.1 General

A GeoAdhoc router shall maintain a local data structure that holds position-related information for the local GeoAdhoc router, i.e. the ego position vector EPV. The data elements of a EPV are specified in clause 8.3.2 and the maintenance of the location table in clause 8.3.3.

8.3.2 Minimum data elements

The EPV shall contain at least the following data elements:

1) Geographical position *POS_EPV*.
2) Speed *S_EPV*.
3) Heading *H_EPV*.
4) Timestamp *TST_EPV* indicating when the geographical position *POS_EPV* was generated.
5) Accuracy of the geographical position *PAI_EPV*.

8.3.3 Maintenance

At start-up, all data elements of the EPV shall be initialized with 0 to indicate an unknown value.

The EPV shall be updated with a frequency of the GN protocol constant `itsGnMinUpdateFrequencyEPV` or higher.

In case of a stationary ITS-S, the timestamp of the EPV shall be updated with a frequency of the GN protocol constant `itsGnMinUpdateFrequencyEPV` or higher. The position field in the EPV shall be set to a value indicating the position of the stationary ITS-S. Speed and heading fields of the EPV shall be set to 0. While the ITS-S is stationary, the value of the position field should not change.

8.4 Sequence number

8.4.1 General

Each GeoAdhoc router shall maintain a local sequence number per GAGH that determines the Sequence Number (SN) field of the next GeoNetworking packet to be transmitted.

8.4.2 Maintenance

For a GAGH, the SN shall be initialized to 0. For every GeoNetworking packet P, the sequence number SN(P) shall be incremented as follows:

$$SN(P) = (SN(P) + 1) \bmod SN\_MAX$$

with SN(P) being the sequence number of the GeoNetworking packet and SN_MAX the largest possible sequence number. The resulting sequence number shall be included in the GeoNetworking packet.

The SN is incremented for multi-hop GeoNetworking packets only. Single-hop GeoNetworking packets (BEACON, SHB) do not carry a *SN* field.

8.5 Location service packet buffer

8.5.1 General

Upon invocation of the LS (clause 10.2.4) by a GAGH, a GeoAdhoc router shall queue a GeoNetworking packet in a *LS packet buffer* for the sought destination until the LS is completed. Subsequent GeoNetworking packets, which are processed while the LS is in progress, shall also be buffered (see clause 10.2.4).

8.5.2 Buffer size

The *LS packet buffer* shall have a minimum size of the value stored in the GN protocol constant `itsGnLocationServicePacketBufferSize`.

8.5.3 Maintenance

The *LS packet buffer* shall work as follows:

1) GeoNetworking packets arriving at the *LS packet buffer* for a destination (GN_ADDR of a certain ITS-S) shall be queued at the tail of the queue.

2) When a new GeoNetworking packet arrives at the *LS packet buffer* and exceeds the buffer capacity (buffer overflow), GeoNetworking packets from the head of the queue are removed and the new GeoNetworking packet queued at the tail (head drop).

3) When the LS is completed, the *LS packet buffer* shall be flushed, i.e. all GeoNetworking packets stored in the buffer shall be sent in a FIFO manner.

4) When the queuing time of the GeoNetworking packet in the *LS packet buffer* exceeds the packet lifetime carried in the GeoNetworking packet's *LT* field in the *Basic Header*, the GeoNetworking packet shall be discarded.

5) When a stored GeoNetworking packet is sent:

   a) the *LT* field shall be reduced by the queuing time in the *LS packet buffer*;

   b) it is recommended to update the SO PV.

NOTE 1: When security is enabled, i.e. the GN protocol constant `itsGnSecurity` is set to ENABLED, and the local GeoAdhoc router is the source of the GeoNetworking packet, the signature may need to be updated. Signatures of forwarded packets are not updated.

6) When the LS does not complete, all stored GeoNetworking packets shall be discarded triggered by the *LS*.

NOTE 2: The mechanism to detect that a LS does not complete is implementation dependent.

8.6 Forwarding packet buffer

8.6.1 General

A GeoAdhoc router shall use *forwarding packet buffers* to temporarily keep packets in a GeoAdhoc router during the forwarding process.

A GeoAdhoc router shall maintain the following *forwarding packet buffers*:

1) *UC forwarding packet buffer* to buffer GUC packets per GN_ADDR per GAGH.

2) *BC forwarding packet buffer* to buffer TSB, GBC and GAC packets per GAGH.

The GeoAdhoc router shall maintain a *CBF packet buffer* if Contention-Based Forwarding (CBF) is enabled, i.e. if:

1) the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 2 (CBF);

2) the GN protocol constant `itsGnAreaForwardingAlgorithm` is set to 2 (CBF) or 3 (ADVANCED).

8.6.2 Buffer size

The *UC forwarding packet buffer* shall have a minimum size given by the value of the GN protocol constant `itsGnUcForwardingPacketBufferSize`.

The *BC forwarding packet buffer* shall have a minimum size given by the value of the GN protocol constant `itsGnBcForwardingPacketBufferSize`.

The *CBF packet buffer* shall have a minimum size given by the value of the GN protocol constant `itsGnCbfPacketBufferSize`.

8.6.3 Maintenance

The *UC forwarding packet buffer* and the *BC forwarding packet buffer* shall work as follows:

1) GeoNetworking packets arriving at the *forwarding packet buffer* shall be queued at the tail of the queue.

2) When a new GeoNetworking packet arrives at the *forwarding packet buffer* and exceeds the buffer capacity, GeoNetworking packets from the head of the queue are removed and the new GeoNetworking packet queued at the tail (head drop).

3) When the *forwarding packet buffer* is flushed, the GeoNetworking packets stored in the buffer shall be forwarded in a FIFO manner.

4) When the queuing time of the GeoNetworking packet in the *forwarding packet buffer* exceeds the packet lifetime carried in the packet's *LT* field in the *Basic Header*, the GeoNetworking packet shall be discarded.

5) When a stored GeoNetworking packet is sent:

   a) the *LT* field shall be reduced by the queuing time in the *forwarding packet buffer*;

   b) it is recommended to update the SO PV.

NOTE 1: When security is enabled, i.e. the GN protocol constant `itsGnSecurity` is set to ENABLED, and the local GeoAdhoc router is the source of the GeoNetworking packet, the signature may need to be updated. Signatures of forwarded packets are not updated.

The *CBF packet buffer* shall work as follows:

1) Packets arriving at the *CBF packet buffer* shall be queued at the tail of the queue.

2) When a new GeoNetworking packet arrives at the *CBF packet buffer* and exceeds the buffer capacity, GeoNetworking packets from the head of the queue are removed and the new GeoNetworking packet queued at the tail (head drop).

3) Every GeoNetworking packet in the buffer is associated with a timer. When the timer expires the GeoNetworking packet is removed from the queue and sent.

4) When a stored GeoNetworking packet is sent:

   a) the *LT* field shall be reduced by the queuing time in the *CBF packet buffer*;

NOTE 2: Due the encoding of the LT field (see clause 9.6.4) the reduction of the LT field by the queuing time may not have an effect.

   b) the SO PV in the sent packet should be updated.

NOTE 3: When security is enabled, i.e. the GN protocol constant `itsGnSecurity` is set to ENABLED, and the local GeoAdhoc router is the source of the GeoNetworking packet, the signature may need to be updated. Signatures of forwarded packets are not updated.

NOTE 4: The value of the timer is set by the CBF forwarding algorithm specified in clause E.3.

9 GeoNetworking packet structure and formats

9.1 Overview

This clause specifies the structure and the format of the GeoNetworking packet.

9.2 Packet structure

9.2.1 General

As specified in ETSI TS 103 836-3 [i.1], the GeoNetworking protocol is either used in the GeoNetworking protocol stack or in the protocol stack that combines the GeoNetworking protocol and IPv6.

9.2.2 Overall packet structure

A GeoNetworking packet is part of the overall frame/packet structure depicted in figure 4 (without security) and figure 6 (with security), respectively:

1) The *Access layer header* is the header of the specific technology at the Access layer with which the packet is transmitted.

NOTE 1: The Access layer header is not specified by the present document. However, the GeoNetworking protocol sets the link layer address, or more generally the link layer address, in order to define and identify the next hop of a GeoNetworking packet.

2) The *GeoNetworking header* is the header of the GeoNetworking packet as defined in the present document.

3) The optional payload represents the user data that are created by upper protocol entities, i.e. the T-SDU or GN6-SDU. It is passed to the GeoNetworking protocol for transmission.

NOTE 2: Some GeoNetworking packets do not carry a payload, such as Beacon.

Access Layer Header	GeoNetworking Header	Payload (optional)

Figure 4: GeoNetworking packet structure (without security)

9.2.3 Maximum Transmit Unit

The Maximum Transmit Unit (MTU), which the GeoNetworking protocol supports via the TRANSP_CORE interface, i.e. the MTU_GN depends on the MTU of the Access layer technology (MTU_ALI) over which the GeoNetworking packet is transported. In particular, MTU_GN shall be less or equal to MTU_ALI reduced by the size of the largest GeoNetworking protocol header (GEO_MAX) including *Basic Header*, *Common Header* and *Extended Header* and security overhead:

$$MTU\_GN \ \leq MTU\_ALI - GEO\_MAX$$

GEO_MAX is set by the GN protocol constant `itsGnMaxGeoNetworkingHeaderSize`.

9.3 GeoNetworking header structure

The GeoNetworking header shall be comprised of a *Basic Header*, *Common Header* and an optional *Extended Header* (figure 5).

Basic Header	Common Header	Extended Header (optional)

Figure 5: GeoNetworking header structure

Basic Header, *Common Header* and *Extended Header* are specified in clause 9.6, clause 10.3.6 and clause 9.8.

NOTE: The composition of the *Basic Header* and *Common Header* equals for all packet transport types and differs for the Extended Header.

9.4 GeoNetworking *Secured Packet*

The overall packet structure may be protected by security services, by digital signatures, certificates, and by encryption.With enabled security (GN protocol constant `itsGnSecurity` is set to ENABLED), the overall packet structure is depicted in figure 6.

Security operations are executed by the security entity via the GAGH_SEC interface (figure 1) and clause 10.3.

Access Layer Header	GeoNetworking Basic Header	GeoNetworking Secured Packet with GeoNetworking Common Header, Optional Extended Header and Optional Payload

Figure 6: GeoNetworking packet structure (with security)

9.5 Position vectors

9.5.1 Overview

For simplicity, a set of position-related fields of the GeoNetworking header are subsumed to a Position Vector (PV). Two types of PV are defined:

1) Long position vector as specified in clause 9.5.2.

2) Short position vector as specified in clause 9.5.3.

9.5.2 *Long Position Vector*

9.5.2.1 Structure

The *Long Position Vector (LPV)* shall consist of the fields specified in figure 7.

<table>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="32">GN_ADDR</td></tr>
<tr><td colspan="32">TST</td></tr>
<tr><td colspan="32">Lat</td></tr>
<tr><td colspan="32">Long</td></tr>
<tr><td>PAI</td><td colspan="15">S</td><td colspan="16">H</td></tr>
</table>

Figure 7: *Long Position Vector*

9.5.2.2 Fields

The *Long Position Vector* (LPV) shall consist of the fields as specified in table 2.

Table 2: Fields of *Long Position Vector*

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	**Last**			
1	*GN_ADDR*	Octet 0	Octet 7	64 bit address	n/a	Network address for the GeoAdhoc router entity in the ITS-S
2	*TST*	Octet 8	Octet 11	32 bit unsigned integer	[ms]	Expresses the time in milliseconds at which the latitude and longitude of the ITS-S were acquired by the GeoAdhoc router. The time is encoded as: $TST = TST(TAI) \bmod 2^{32}$ where *TST(TAI)* is the number of elapsed TAI milliseconds since 2004-01-01 00:00:00.000 UTC
3	*Lat*	Octet 12	Octet 15	32 bit signed integer	[1/10 micro-degree]	WGS 84 [i.3] latitude and longitude of the GeoAdhoc router reference position expressed in 1/10 micro degree
4	*Long*	Octet 16	Octet 19	32 bit signed integer	[1/10 micro-degree]	
5	*PAI*	Octet 20 Bit 0	Octet 20 Bit 0	1 bit unsigned integer	n/a	Position accuracy indicator of the GeoAdhoc router reference position Set to 1 (i.e. True) if the semiMajorConfidence of the PosConfidenceEllipse as specified in ETSI TS 102 894-2 [2] is smaller than the GN protocol constant `itsGnPaiInterval` / 2 Set to 0 (i.e. False) otherwise
6	*S*	Octet 20 Bit 1	Octet 21	15 bit signed integer	[1/100 m/s]	Speed of the GeoAdhoc router expressed in signed units of 0,01 metre per second
7	*H*	Octet 22	Octet 23	16 bit unsigned integer	[1/10 degrees]	Heading of the GeoAdhoc router, expressed in unsigned units of 0,1 degree from North

9.5.3 *Short Position Vector*

9.5.3.1 Structure

The *Short Position Vector* (SPV) shall consist of the fields specified in figure 8.

0								1								2								3							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
GN_ADDR																															
TST																															
Lat																															
Long																															

Figure 8: *Short Position Vector*

9.5.3.2 Fields

The *Short Position Vector* (SPV) shall consist of the fields as specified in table 3.

Table 3: Fields of *Short Position Vector*

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*GN_ADDR*	Octet 0	Octet 7	64 bit address	n/a	*GN_ADDR* field as specified in table 2
2	*TST*	Octet 8	Octet 11	32 bit unsigned integer	[ms]	Timestamp TST field as specified in table 2
3	*Lat*	Octet 12	Octet 15	32 bit signed integer	[1/10 micro-degree]	Latitude (*Lat*) field as specified in table 2
4	*Long*	Octet 16	Octet 19	32 bit signed integer	[1/10 micro-degree]	Longitude (*Long*) field as specified in table 2
NOTE: The timestamp TST field indicates the time when the position (LAT, LONG) of the SPV was acquired.						

9.6 *Basic Header*

9.6.1 Composition of the *Basic Header*

The *Basic Header* shall be present in every GeoNetworking packet and consists of the fields as depicted in figure 9.

0								1								2								3							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
Version				NH				Reserved								LT								RHL							

Figure 9: *Basic Header* format

9.6.2 Fields of the *Basic Header*

The *Basic Header* shall carry the fields as specified in table 4.

Table 4: Fields of the *Basic Header*

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*Version*	Octet 0 Bit 0	Octet 0 Bit 3	4 bit unsigned integer	n/a	Identifies the version of the GeoNetworking protocol
2	*NH*	Octet 0 Bit 4	Octet 0 Bit 7	4 bit unsigned integer	n/a	Identifies the type of header immediately following the GeoNetworking *Basic Header* as specified in table 5
3	*Reserved*	Octet 1	Octet 1	8-bit unsigned integer	n/a	Reserved Set to 0
4	*LT*	Octet 2	Octet 2	8 bit unsigned integer	n/a	Lifetime field. Indicates the maximum tolerable time a packet may be buffered until it reaches its destination Bit 0 to Bit 5: LT sub-field Multiplier Bit 6 to Bit 7: LT sub-field Base Encoded as specified in clause 9.6.4
5	*RHL*	Octet 3	Octet 3	8 bit unsigned integer	[hops]	Decremented by 1 by each GeoAdhoc router that forwards the packet The packet shall not be forwarded if *RHL* is decremented to zero

9.6.3 Encoding of the *NH* field in the *Basic Header*

For the *Next Header (NH)* field in the *Basic Header* the values as specified in table 5 shall be used.

Table 5: *Next Header* (*NH*) field in the GeoNetworking *Basic Header*

Next Header (NH)	Encoding	Description
ANY	0	Unspecified
Common Header	1	GeoNetworking *Common Header* as specified in clause 10.3.6
Secured Packet	2	GeoNetworking *Secured Packet* as specified in clause 9.4
NOTE: The *Common Header* also carries a *NH* field.		

9.6.4 Encoding of the LT field

The *Lifetime (LT)* field shall indicate the maximum tolerable time a packet may be buffered until it reaches its destination.

NOTE 1: This parameter is relevant for safety and traffic efficiency information that do not have strict real-time requirements. In sparse network scenarios, this lifetime may also be used to avoid re-transmission and forwarding of outdated information.

NOTE 2: When a GeoNetworking packet is buffered, the value of the *Lifetime (LT)* field is reduced by the queuing time in the packet buffer.

The following method for encoding of the *LT* field uses a non-linear encoding, which provides a high resolution for low numbers and progressively lower resolution for higher numbers.

The *LT* field shall be comprised of two sub-fields: a $LT_{Multiplier}$ sub-field (*Multiplier*) and a LT_{Base} sub-field (*Base*) (figure 10) and shall be encoded as follows:

$$Lifetime_{decoded} = LT_{Multiplier} \times T_{Base}$$

The LT_{Base} sub-field represents a two bit unsigned selector that chooses one out of four predefined values as specified in table 6.

Table 6: Encoding of *LT* sub-field *LT Base*

Value	LT_{base}
0	50 ms
1	1 s
2	10 s
3	100 s

The $LT_{Multiplier}$ is a 6 bit unsigned integer, which represents a multiplier range from 0 to $2^6 - 1 = 63$.

The default value of the LT field is set to the GN protocol constant `itsGnDefaultPacketLifetime`. The value shall be smaller than the GN protocol constant `itsGnMaxPacketLifetime`.

0	1	2	3	4	5	6	7
Multiplier [0 to 63]						Base: 50 ms, 1 s, 10 s, 100 s	

Figure 10: Composition of the *LT field*

9.7 *Common Header*

9.7.1 Composition of the *Common Header*

The *Common Header* shall be present in every GeoNetworking packet and consists of the fields as depicted in figure 11.

0								1								2								3							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
NH				Reserved				HT				HST				TC								Flags							
PL																MHL								Reserved							

Figure 11: *Common Header* format

9.7.2 Fields of the *Common Header*

The *Common Header* shall carry the fields as specified in table 7.

Table 7: Fields of the *Common Header*

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*NH*	Octet 0 Bit 0	Octet 0 Bit 3	4 bit unsigned integer	n/a	Identifies the type of header immediately following the GeoNetworking headers as specified in table 8
2	*Reserved*	Octet 0 Bit 4	Octet 0 Bit 7	4 bit unsigned integer	n/a	Reserved Set to 0
3	*HT*	Octet 1 Bit 0	Octet 1 Bit 3	4 bit unsigned integer	n/a	Identifies the type of the GeoNetworking header as specified in table 9
4	*HST*	Octet 1 Bit 4	Octet 1 Bit 7	4 bit unsigned integer	n/a	Identifies the sub-type of the GeoNetworking header as specified in table 9
5	*TC*	Octet 2	Octet 2	8 bit unsigned integer	n/a	Traffic class that represents Facilities layer requirements on packet transport. Encoding is specified in clause 9.7.5
6	*Flags*	Octet 3	Octet 3	Bit field	n/a	Bit 0: Indicates whether the ITS-S is mobile or stationary (GN protocol constant `itsGnIsMobile`) Bit 1 to Bit 7: Reserve, set to 0

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
7	*PL*	Octet 4	Octet 5	16 bit unsigned integer	[octets]	Length of the GeoNetworking payload, i.e. the rest of the packet following the whole GeoNetworking header in octets, for example BTP + CAM
8	*MHL*	Octet 6	Octet 6	8 bit unsigned integer	[hops]	Maximum hop limit (see note)
9	*Reserved*	Octet 7	Octet 7	8 bit unsigned integer	n/a	Reserved Set to 0
NOTE: The Maximum hop limit is not decremented by a GeoAdhoc router that forwards the packet.						

9.7.3 Encoding of the *NH* field in the *Common Header*

For the *Next Header (NH)* field in the *Common Header* the values as specified in table 8 shall be used.

Table 8: *Next Header* (*NH*) field in the GeoNetworking *Common Header*

Next Header (NH)	Encoding	Description
ANY	0	Unspecified
BTP-A	1	Transport protocol (BTP-A for interactive packet transport), see ETSI TS 103 836-5-1 [3]
BTP-B	2	Transport protocol (BTP-B for non-interactive packet transport), see ETSI TS 103 836-5-1 [3]
IPv6	3	IPv6 header, see ETSI TS 103 836-3 [i.1]

NOTE: The *Basic Header* also carries a *NH* field.

9.7.4 Encoding of the *HT* and *HST* fields

For the *Header Type* (*HT)* and the *Header Sub-Type* (*HST)* fields in the *Common Header* the values as specified in table 9 shall be used.

Table 9: GeoNetworking *Header Types* and *Header Sub-Types*

Header Type (HT)	Header Sub-Type (HST)	Encoding	Description
ANY		0	Unspecified
	UNSPECIFIED	0	Unspecified
BEACON		1	Beacon
	UNSPECIFIED	0	Unspecified
GEOUNICAST		2	GeoUnicast
	UNSPECIFIED	0	Unspecified
GEOANYCAST		3	Geographically-Scoped Anycast (GAC)
	GEOANYCAST_CIRCLE	0	Circular area
	GEOANYCAST_RECT	1	Rectangular area
	GEOANYCAST_ELIP	2	Ellipsoidal area
GEOBROADCAST		4	Geographically-Scoped Broadcast (GBC)
	GEOBROADCAST_CIRCLE	0	Circular area
	GEOBROADCAST_RECT	1	Rectangular area
	GEOBROADCAST_ELIP	2	Ellipsoidal area
TSB		5	Topologically-Scoped Broadcast (TSB)
	SINGLE_HOP	0	Single-Hop Broadcast (SHB)
	MULTI_HOP	1	Multi-hop TSB
LS		6	Location service (LS)
	LS_REQUEST	0	Location service request
	LS_REPLY	1	Location service reply

9.7.5 Encoding of the *TC* field

The *TC* field shall consist of the fields as depicted in figure 12.

0	1	2	3	4	5	6	7
SCF	Reserved	TC ID					

Figure 12: *Traffic Class (TC)* field composition

Table 10: *TC* field in the GeoNetworking *Common Header*

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*SCF*	Bit 0	Bit 0	Bit	n/a	Indicates whether the packet shall be buffered when no suitable neighbour exists Store-Carry-Forward (SCF) Length: 1 bit
2	*Reserved*	Bit 1	Bit 1	Bit	n/a	Reserved. Length: 1 bit
3	*TC ID*	Bit 2	Bit 7	6-bit unsigned integer	n/a	TC ID as specified in the media-dependent part of GeoNetworking corresponding to the interface over which the packet will be transmitted. Length: 6 bits

The default value for the *TC* field is set by the GN protocol constant `itsGnDefaultTrafficClass`.

9.8 GeoNetworking packet header types

9.8.1 Overview

The following GeoNetworking packet header types are defined:

1) GUC packet header (clause 9.8.2).

2) TSB packet header (clause 9.8.3).

3) SHB packet header (clause 9.8.4).

4) GBC and GAC packet headers (clause 9.8.5).

5) BEACON packet header (clause 9.8.6).

6) LS Request and LS Reply packet headers (clause 9.8.7 and clause 9.8.8).

9.8.2 GUC packet header

9.8.2.1 Composition of the GUC packet header

The GUC header shall be comprised of the *Basic Header*, the *Common Header* and the *Extended Header* as shown in figure 13.

NOTE: The *Extended Header* comprises all fields except the *Basic Header* and the *Common Header*.

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="32">Basic Header</td></tr>
<tr><td colspan="32">Common Header</td></tr>
<tr><td colspan="16">SN</td><td colspan="16">Reserved</td></tr>
<tr><td colspan="32">SO PV</td></tr>
<tr><td colspan="32">DE PV</td></tr>
</table>

Figure 13: Packet header format: GUC

9.8.2.2 Fields of the GUC packet header

The GUC packet header shall consist of the fields as specified in table 11.

Table 11: Fields of the GUC packet header

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	**Last**			
1	*Basic Header*	Octet 0	Octet 3	Basic Header	n/a	*Basic Header* as specified in clause 9.6 Length: 4 octets
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 10.3.6 Length: 8 octets
3	*SN*	Octet 12	Octet 13	16-bit unsigned integer	n/a	Sequence number field. Indicates the index of the sent GUC packet (clause 8.4) and used to detect duplicate GeoNetworking packets (annex A)
4	*Reserved*	Octet 14	Octet 15	16-bit unsigned integer	n/a	Reserved Set to 0
5	*SO PV*	Octet 16	Octet 39	Long position vector	n/a	*Long Position Vector* (LPV) containing the reference position of the source as specified in clause 9.5.2 Length: 24 octets
6	*DE PV*	Octet 40	Octet 59	Short position vector	n/a	*Short Position Vector* (SPV) containing the position of the destination. It shall consist of the fields as specified in clause 9.5.3 Length: 20 octets

9.8.3 TSB packet header

9.8.3.1 Composition of the TSB packet header

The TSB header shall be comprised of the *Basic Header*, the *Common Header* and the *Extended Header* as shown in figure 14.

NOTE: The *Extended Header* comprises all fields except the *Basic Header* and the *Common Header*.

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="32">Basic Header</td></tr>
<tr><td colspan="32">Common Header</td></tr>
<tr><td colspan="16">SN</td><td colspan="16">Reserved</td></tr>
<tr><td colspan="32">SO PV</td></tr>
</table>

Figure 14: Packet header format: TSB

9.8.3.2 Fields of the TSB packet header

The TSB packet header shall consist of the fields as specified in table 12.

Table 12: Fields of the TSB packet header

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*Basic Header*	Octet 0	Octet 3	Basic Header	n/a	*Basic Header* as specified in clause 9.6 Length: 4 octets
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 10.3.6 Length: 8 octets
3	*SN*	Octet 12	Octet 13	16-bit unsigned integer	n/a	Sequence number field. Indicates the index of the sent TSB packet (clause 8.4) and used to detect duplicate GeoNetworking packets (annex A)
4	*Reserved*	Octet 14	Octet 15	16-bit unsigned integer	n/a	Reserved Set to 0
5	*SO PV*	Octet 16	Octet 39	Long Position Vector	n/a	*Long Position Vector* containing the reference position of the source as specified in clause 9.5.2 (*Long Position Vector*) Length: 24 octets

9.8.4 SHB packet header

9.8.4.1 Composition of the SHB packet header

The SHB header shall consist of the *Basic Header*, the *Common Header* and the *Extended Header* as shown in figure 15.

NOTE: The *Extended Header* comprises all fields except the *Basic Header* and the *Common Header*.

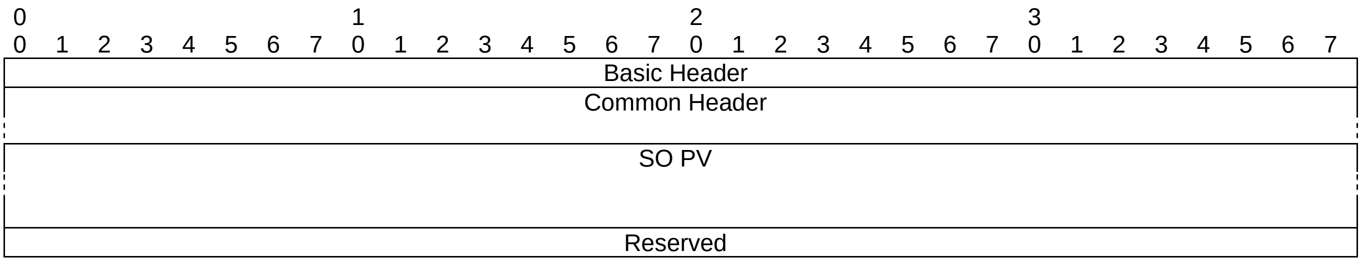

Figure 15: Packet header format: SHB

9.8.4.2 Fields of the SHB packet header

The SHB packet header shall consist of the fields as specified in table 13.

Table 13: Fields of the SHB packet header

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	**Last**			
1	*Basic Header*	Octet 0	Octet 3	Basic Header	n/a	*Basic Header* as specified in clause 9.6 Length: 4 octets
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 10.3.6 Length: 8 octets
3	*SO PV*	Octet 12	Octet 35	Long Position Vector	n/a	*Long Position Vector* containing the reference position of the source. It shall carry the fields as specified in clause 9.5.2 (*Long Position Vector*) Length: 24 octets
4	*Media-dependent data*	Octet 36	Octet 39	32-bit unsigned integer	n/a	Used for media-dependent operations If not used, it shall be set to 0

9.8.5 GBC/GAC packet header

9.8.5.1 Composition of the GBC/GAC packet header

The GBC and GAC packets shall have the same header structure. They are distinguished by the *HT* field in the *Common Header*.

The header shall be comprised of the *Basic Header*, the *Common Header* and the *Extended Header* as shown in figure 16.

NOTE: The *Extended Header* comprises all fields except the *Basic Header* and the *Common Header*.

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><th>0 1 2 3 4 5 6 7</th><th>0 1 2 3 4 5 6 7</th><th>0 1 2 3 4 5 6 7</th><th>0 1 2 3 4 5 6 7</th></tr>
<tr><td colspan="4">Basic Header</td></tr>
<tr><td colspan="4">Common Header</td></tr>
<tr><td colspan="2">SN</td><td colspan="2">Reserved</td></tr>
<tr><td colspan="4">SO PV</td></tr>
<tr><td colspan="4">GeoAreaPos Latitude</td></tr>
<tr><td colspan="4">GeoAreaPos Longitude</td></tr>
<tr><td colspan="2">Distance a</td><td colspan="2">Distance b</td></tr>
<tr><td colspan="2">Angle</td><td colspan="2">Reserved</td></tr>
</table>

Figure 16: Packet header format: GBC/GAC

9.8.5.2 Fields of the GBC/GAC packet header

The GBC/GAC packet header shall consist of the fields as specified in table 14.

Table 14: Fields of the GBC/GAC packet header

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*Basic Header*	Octet 0	Octet 3	Basic Header	n/a	*Basic Header* as specified in clause 9.6 Length: 4 octets
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 10.3.6 Length: 8 octets
3	*SN*	Octet 12	Octet 13	16-bit unsigned integer	n/a	Sequence number field. Indicates the index of the sent GBC/GAC packet (clause 8.4) and used to detect duplicate GeoNetworking packets (annex A)
4	*Reserved*	Octet 14	Octet 15	16-bit unsigned integer	n/a	Reserved Set to 0
5	*SO PV*	Octet 16	Octet 39	Long position vector	n/a	*Long Position Vector* containing the reference position of the source as specified in clause 9.5.2 (*Long Position Vector*) Length: 24 octets
6	*GeoAreaPos Latitude*	Octet 40	Octet 43	32-bit signed integer	[1/10 micro-degree]	WGS 84 [i.3] latitude for the centre position of the geometric shape as defined in ETSI TS 103 899 [1] in 1/10 micro degree
7	*GeoAreaPos Longitude*	Octet 44	Octet 47	32-bit signed integer	[1/10 micro-degree]	WGS 84 [i.3] longitude for the centre position of the geometric shape as defined in ETSI TS 103 899 [1] in 1/10 micro degree
8	*Distance a*	Octet 48	Octet 49	16-bit unsigned integer	[m]	Distance a of the geometric shape as defined in ETSI TS 103 899 [1] in metres
9	*Distance b*	Octet 50	Octet 51	16-bit unsigned integer	[m]	Distance b of the geometric shape as defined in ETSI TS 103 899 [1] in metres
10	*Angle*	Octet 52	Octet 53	16-bit unsigned integer	[°]	Angle of the geometric shape as defined in ETSI TS 103 899 [1] in degrees from North
11	*Reserved*	Octet 54	Octet 55	16-bit unsigned integer	n/a	Reserved Set to 0

In case of a circular area (GeoNetworking packet sub-type *HST* = 0), the fields shall be set to the following values:

1) *Distance a* is set to the radius r.

2) *Distance b* is set to 0.

3) *Angle* is set to 0.

9.8.6 BEACON packet header

9.8.6.1 Composition of the BEACON packet header

A BEACON packet shall consist of the *Basic Header*, the *Common Header*, and the *Extended Header* as shown in figure 17.

NOTE: The *Extended Header* comprises all fields except the *Basic Header* and the *Common Header*.

0								1								2								3							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7

Basic Header

Common Header

SO PV

Figure 17: Packet header format: BEACON

9.8.6.2 Fields of the BEACON packet header

The BEACON shall consist of the fields of the *Basic Header*, the *Common Header* and the *Extended Header* as specified in table 15.

Table 15: Fields of the BEACON packet header

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*Basic Header*	Octet 0	Octet 3	Basic Header	n/a	*Basic Header* as specified in clause 9.6 Length: 4 octets
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common header* as specified in clause 10.3.6 Length: 8 octets
3	*SO PV*	Octet 12	Octet 35	Long Position Vector	n/a	*Long Position Vector* containing the reference position of the source. It shall carry the fields as specified in clause 9.5.2 (*Long Position Vector*) Length: 24 octets

9.8.7 LS Request packet header

9.8.7.1 Composition of the LS Request packet header

The LS Request packet header shall be comprised of the *Common Header* and the *Extended Header* as shown in figure 18.

NOTE: The *Extended Header* comprises all fields except the *Basic Header* and the *Common Header*.

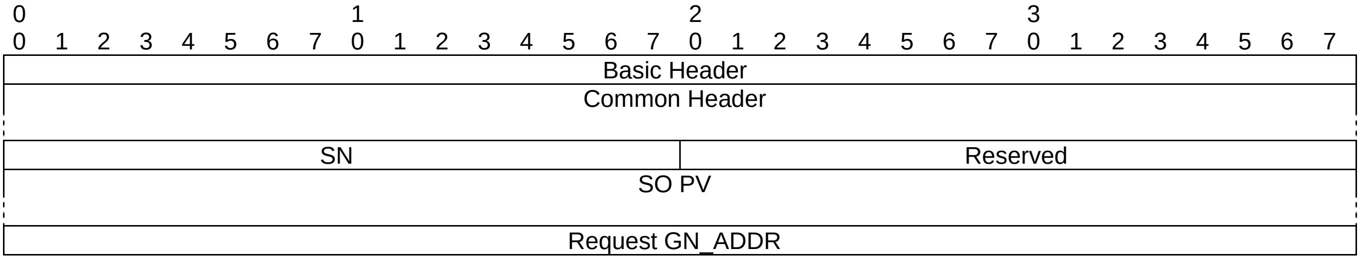

Figure 18: Packet header format: LS Request

9.8.7.2 Fields of the LS Request packet header

The LS Request packet header shall carry the fields as specified in table 16.

Table 16: Fields of the LS Request packet header

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*Basic Header*	Octet 0	Octet 3	Basic Header	n/a	*Basic Header* as specified in clause 9.6 Length: 4 octets
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 10.3.6 Length: 8 octets
3	*SN*	Octet 12	Octet 13	16-bit unsigned integer	n/a	Sequence number field. Indicates the index of the sent LS Request packet (clause 8.4) and used to detect duplicate GeoNetworking packets (annex A)
4	*Reserved*	Octet 14	Octet 15	16-bit unsigned integer	n/a	Reserved Set to 0
5	*SO PV*	Octet 16	Octet 39	Long position vector	n/a	*Long Position Vector* containing the position of the source as specified in clause 9.5.2 (*Long Position Vector*) Length: 24 octets
6	*Request GN_ADDR*	Octet 40	Octet 47	64-bit address	n/a	The GN_ADDR address for the GeoAdhoc router entity for which the location is being requested

9.8.8 LS Reply packet header

9.8.8.1 Composition of the LS Reply packet header

The LS Reply packet header shall be comprised of the *Basic Header*, the *Common Header* and the *Extended Header* as shown in figure 19.

NOTE: The *Extended Header* comprises all fields except the *Basic Header* and the *Common Header*.

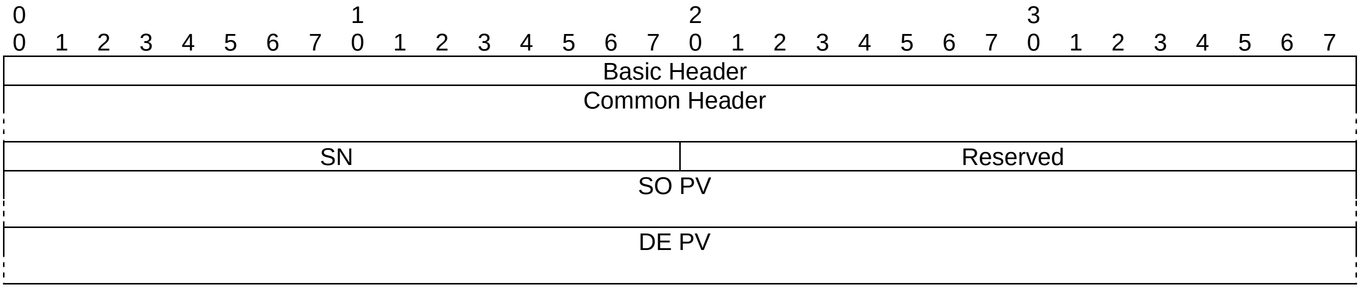

Figure 19: Packet header format: LS Reply

9.8.8.2 Fields of the LS Reply packet header

The LS Reply packet header shall carry the fields as specified in table 17.

Table 17: Fields of the LS Reply packet header

Field #	Field name	Octet/bit position		Type	Unit	Description
		First	Last			
1	*Basic Header*	Octet 0	Octet 3	Basic Header	n/a	*Basic Header* as specified in clause 9.6 Length: 4 octets
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 10.3.6 Length: 8 octets
3	*SN*	Octet 12	Octet 13	16-bit unsigned integer	n/a	Sequence number field. Indicates the index of the sent LS Reply packet (clause 8.4) and used to detect duplicate GeoNetworking packets (annex A)
4	*Reserved*	Octet 14	Octet 15	16-bit unsigned integer	n/a	Reserved Set to 0
5	*SO PV*	Octet 16	Octet 39	Long position vector	n/a	*Long Position Vector* containing the reference position of the source, which represents the Request GN_ADDR in the corresponding LS Request, as specified in clause 9.5.2 (*Long Position Vector*) Length: 24 octets
6	*DE PV*	Octet 40	Octet 59	Short position vector	n/a	*Short Position Vector* containing the reference position of the destination. It shall carry the fields as specified in clause 9.5.3 (*Short Position Vector*) Length: 20 octets

10 Protocol operation

10.1 General

This clause specifies the media-independent operations of the GeoNetworking protocol.

The operations include:

1) Network management:
   - Address configuration (clause 10.2.1);
   - Local position vector and time update (clause 10.2.2);
   - Beaconing (clause 10.2.3);
   - Location service (clause 10.2.4).
2) Packet handling:
   - GUC (clause 10.3.8);
   - TSB (clause 10.3.9);
   - SHB (clause 10.3.10);
   - GBC (clause 10.3.11);
   - GAC (clause 10.3.12).

3) MCO management:

- MCO parameters updates (clause 10.4.2);
- Network updates (clause 10.4.3);
- Handling of packets without assigned ALI (clause 10.4.4).

10.2 Network management

10.2.0 Overview

In the Management plane of the Networking & Transport layer, network management for the GeoNetworking protocol is handled by the GN Core and the GAGHs as depicted in figure 20. The GN Core provides the GeoNetworking address and current local position and time information upon request from a GAGH as specified in clause 10.2.1 and clause 10.2.2, respectively.

Each GAGH applies the duplicate address detection mechanism as specified in clause 10.2.1.5 performs beaconing, and realizes LS operations as specified in clause 10.2.3 and clause 10.2.4, respectively.

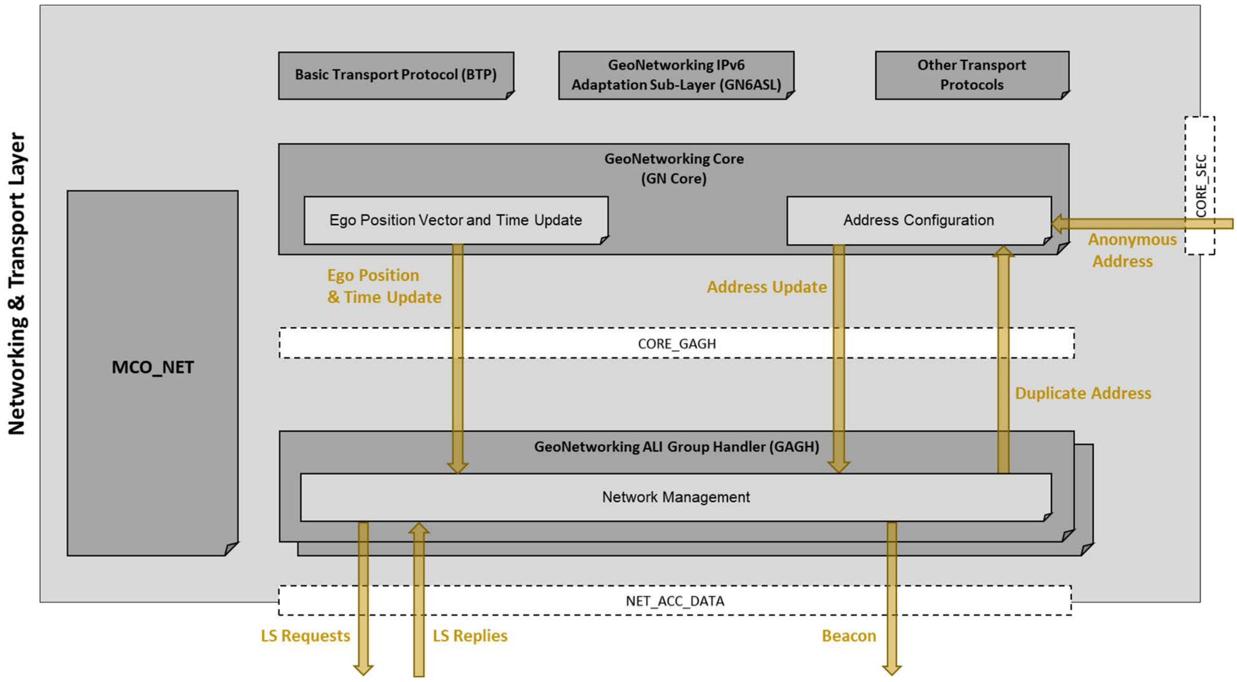

Figure 20: Overview of Network Management functions and control flow

10.2.1 Address configuration

10.2.1.1 General

When initialized, the GN Core shall have a self-assigned initial GeoNetworking address with the format specified in clause 6. GeoNetworking defines three methods for the configuration of the local GN_ADDR:

1) Auto-address configuration (clause 10.2.1.2);
2) Managed address configuration (clause 10.2.1.3);
3) Anonymous address configuration (clause 10.2.1.4).

The method is defined in the GN protocol constant `itsGnLocalAddrConfMethod`.

In the auto-address configuration, the GeoNetworking address cannot be changed. In the managed address configuration, the initial GeoNetworking address (clause 10.2.1.3.2) can be updated (clause 10.2.1.3.3) by the GN Core. In the anonymous address configuration, the address is configured by the security entity and updated to the GAGHs by the GN Core.

Operations for duplicate address detection (DAD) are specified in clause 10.2.1.5. DAD is applied for auto-address configuration (clause 10.2.1.2) and managed address configuration (clause 10.2.1.3).

NOTE: For multi-channel operation, an ITS-S with several GAGHs uses a single GeoNetworking address (see clause 6 "GeoNetworking address").

10.2.1.2 Auto-address configuration

The auto-address configuration method shall be used if the GN protocol constant `itsGnLocalAddrConfMethod` is set to AUTO (0).

When activated, a GAGH shall assign the MID field of the local GN_ADDR from the GN protocol constant `itsGnLocalGnAddr`.

NOTE: The setting of the GN protocol constant `itsGnLocalGnAddr` is implementation dependent. One example implementation is the usage of randomly-generated addresses.

The local GN_ADDR shall not be changed unless the GN protocol constant `itsGnLocalAddrConfMethod` is set to MANAGED (1) or ANONYMOUS (2) or when DAD is invoked.

10.2.1.3 Managed address configuration

10.2.1.3.1 General Requirements

The managed address configuration method shall be used if the GN protocol constant `itsGnLocalAddrConfMethod` is set to MANAGED (1).

With managed address configuration, the GN Core is responsible for providing the MID field of the GN_ADDR.

10.2.1.3.2 Initial address configuration

At startup, a GAGH shall request an MID field for the GN_ADDR from the GN Core (see clause K.2, service primitive *CORE_GAGH.request*). The GN Core is responsible for generating the appropriate GeoNetworking address (see clause K.3 using the service primitive *CORE_GAGH.response*).

10.2.1.3.3 Address update

The update of the MID field of the local GN_ADDR may be triggered by the GN Core using an unsolicited *CORE_GAGH.response* to the active GAGHs.

Upon reception of the *CORE_GAGH.response*, a GAGH shall update its local GN_ADDR.

NOTE 1: For privacy reasons, the GN_ADDR may be derived from the current authorization ticket. The frequency of update and the algorithm of generating pseudonyms are beyond the scope of the present document.

NOTE 2: From communication point of view, a frequent update of the GN_ADDR may impair the performance of the GeoNetworking protocol.

10.2.1.4 Anonymous address configuration

The anonymous address configuration method shall be used if the GN protocol constant `itsGnLocalAddrConfMethod` is set to ANONYMOUS (2). This method allows for configuration of anonymous addresses controlled by the security entity. The services are provided via the CORE_SEC interface.

In this method, the GN Core subscribes to the *ID-CHANGE-SUBSCRIBE* service at the security entity. In one possible implementation, it may register a callback function, which is executed when the pseudonym is changed.

At startup, the GN Core shall execute the following operations:

1) subscribe to the *IDCHANGE-SUBSCRIBE* service provided by the security entity and send a service primitive *SN-IDCHANGE-SUBSCRIBE.request*;

2) process the service primitive *SN-IDCHANGE-SUBSCRIBE.confirm* that returns the subscription handle;

3) process the service primitive *SN-IDCHANGE-EVENT.indication* that provides the parameter *id*. The GN Core shall update the GeoNetworking Address of the active GAGHs using an unsolicited *CORE_GAGH.response*;

4) generate the service primitive *SN-IDCHANGE-EVENT.response* to acknowledge the given command.

When the GeoAdhoc router is shutdown or restarted, the GN Core should execute the following operations:

1) unsubscribe from the *IDCHANGE-SUBSCRIBE* service provided by the security entity and send a service primitive *SN-IDCHANGE-UNSUBSCRIBE.request*.

10.2.1.5 Duplicate address detection

In order to achieve uniqueness of the GeoNetworking address configuration with the auto-address configuration method, i.e. if the GN protocol constant `itsGnLocalAddrConfMethod` is set to AUTO (0), each GAGH shall execute the following operations for DAD:

1) Upon reception of a GeoNetworking packet, a GAGH compares:

   a) its local GN_ADDR and the GN_ADDR of the SO carried in the GeoNetworking packet header; and

   b) its local link layer address (i.e. the MID field of the GN_ADDR, clause 6), with the sender link layer address of the frame.

2) If a conflict is detected, the GAGH shall request a new MID field for the GeoNetworking address from the GN Core entity using the service primitive *CORE_GAGH.request* (clause K.2) indicating *Duplicate address* as the *Request cause*.

3) In case of conflict, the GN Core shall update the new GN_ADDR to all active GAGHs using an unsolicited *CORE_GAGH.response*.

NOTE: In case the GN MID changes, the link layer address also changes. In one possible implementation the link layer address is set from the GAGHs.

10.2.2 Ego position vector and time update

10.2.2.1 Overview

Ego position vector and time are set by the GN Core for the GAGH via the CORE_GAGH interface (clause K.3).

10.2.2.2 Ego Position Vector update

For position update, the GN Core shall send an unsolicited *CORE_GAGH.response* with the *Ego position vector* parameter (clause K.3) to the GAGHs. Upon reception of the *CORE_GAGH.response* with the *Ego position vector* parameter, the GeoAdhoc router updates its EPV (clause 8.3).

As specified in clause 8.3.3, the EPV shall be updated with a minimum frequency of the GN protocol constant `itsGnMinUpdateFrequencyEPV`.

NOTE: The ego position vector is not GAGH-specific.

10.2.2.3 Time update

For time update, the GN_CORE entity sends an unsolicited *CORE_GAGH.response* with the *Time* parameter (clause K.3) to the GeoAdhoc router. Upon reception of the *CORE_GAGH.response* with the *Time* parameter, the GeoAdhoc router should set its local system time at a reasonable time interval.

It should be noted that time management shall support TAI.

NOTE 1: The time is not GAGH-specific.

NOTE 2: Details of the system time management and usage are implementation specific.

10.2.3 Beaconing

Beaconing is used to periodically advertise a GeoAdhoc router's position vector to its neighbours.

Each active GAGH shall send a BEACON periodically unless the GAGH dissiminated another GeoNetworking packet that carries the GeoAdhoc router's EPV. At startup a GeoAdhoc router shall send an initial beacon to announce its presence with every active GAGHs to other GeoAdhoc routers.

NOTE 1: In one possible implementation a timer schedules the transmission of BEACON packets and is reset upon transmission of a GeoNetworking packet that carries a LPV, i.e. a SHB packet.

NOTE 2: Per each GAGH, a default ALI to disseminate the BEACONS may be configured.

10.2.4 Location service

The location service is used if a GeoAdhoc router needs to determine the position of another GeoAdhoc router using one of its GAGH. This is the case if a GeoAdhoc router is in the process to send a T/GN6-SDU as a GUC packet to another GeoAdhoc router, i.e. from the source to the destination, and does not have the position information for the destination in its LocT.

The execution of a location service is fully transparent to protocol entities of higher layers and resides on top of the forwarding function.

The location service is based on the exchange of control packets between GeoAdhoc routers (figure 21). The querying GeoAdhoc router (source) issues a LS Request packet with the GN_ADDR of the sought GeoAdhoc router (destination). The LS Request packet is forwarded by intermediate GeoAdhoc routers (forwarders) until it reaches the destination. The destination replies with a LS Reply packet.

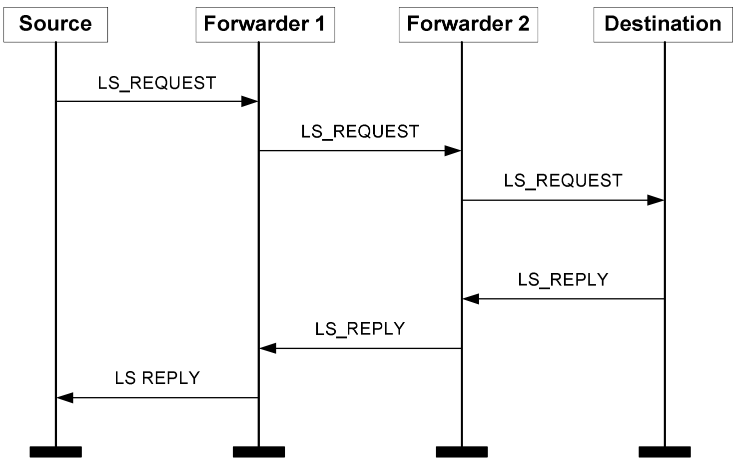

Figure 21: Message sequence chart for the location service (example scenario with two forwarders)

10.3 Packet handling

10.3.1 Overview

This clause defines the behaviour of the protocol in the source, forwarder and destination. Packet handling includes the procedures to determine the destination (GeoAdhoc router, geographical area) of the GeoNetworking packet, execute security functions, execute functions that are specific to the packet type and the associated ALI group, and pass the packet to the ALI via the NET_ACC_DATA interface (figure 22).

The following GN packet types are handled by every GAGH:

- Beacon packet (clause 10.3.6);
- LS packet (clause 10.3.7);
- GUC packet (clause 10.3.8);
- TSB packet (clause 10.3.9);
- SHB packet (clause 10.3.10);
- GBC packet (clause 10.3.11); and
- GAC packet (clause 10.3.12).

The packet handling is further specified in the following clauses.

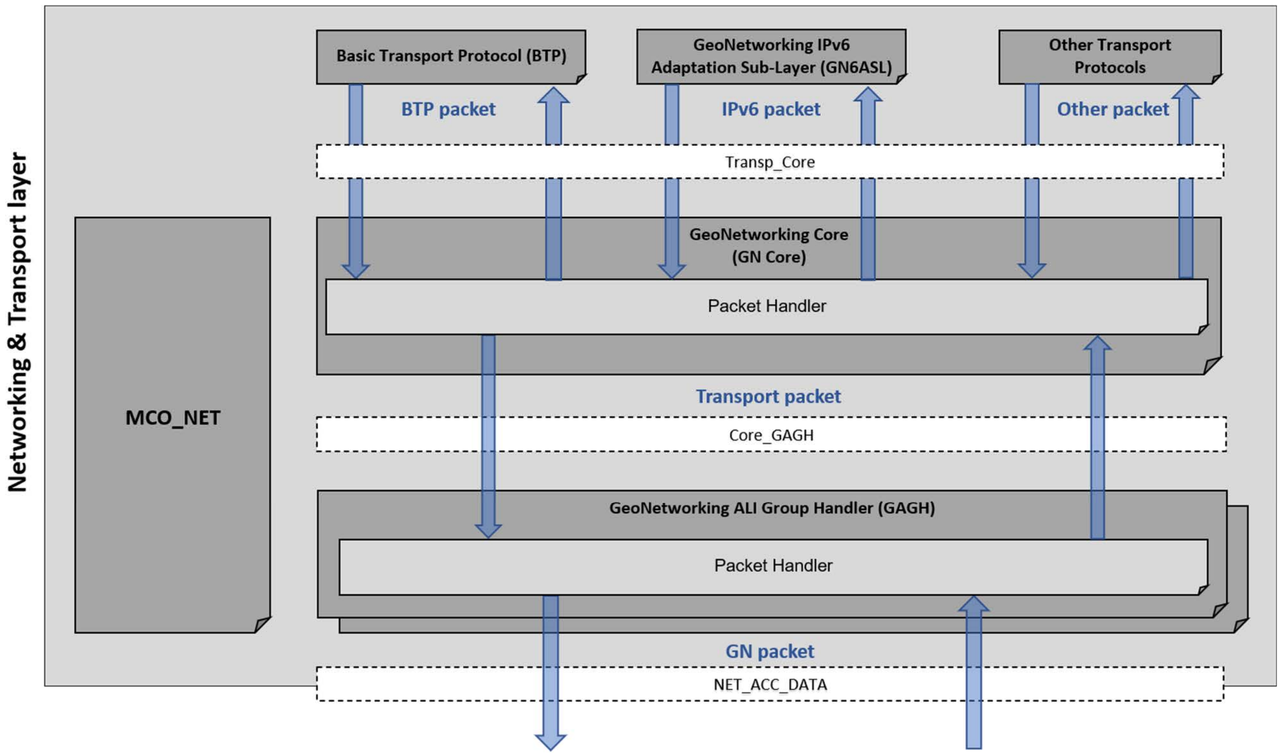

Figure 22: GeoNetworking packet handling

10.3.2 *Basic Header* field settings

For all GeoNetworking packets, the fields of the *Basic Header* shall be set as specified in table 18.

Table 18: Field settings for the *Basic Header*

Field name	Field setting	Description
Version	GN protocol constant `itsGnProtocolVersion`	Version of the GeoNetworking protocol
NH	1 (*Common Header*) if *TRANSP_CORE.request* parameter *Security profile* indicates that the packet is unsecured. 2 (*Secured Packet*) if *TRANSP_CORE.request* parameter *Security profile* indicates that the packet is secured. For *Header type* HT = 1 (BEACON) set the value of the *NH* field to 1 (Common Header) if GN protocol constant `itsGnSecurity` is DISABLED or to 2 (Secured Header) if it is ENABLED	Next header (table 5 in clause 9.6.3)
Reserved	Set to 0	Reserved Set to 0
LT	Source: • Value of optional *Maximum packet lifetime* parameter from service primitive *TRANSP_CORE.request, or* • GN protocol constant `itsGnDefaultPacketLifetime` if the *Maximum packet lifetime* parameter is not set, or if *Header type* HT = 1 (BEACON) Forwarder: When the GeoNetworking packet is stored in a packet buffer, the value of the received LT field shall be reduced by the queuing time in the packet buffer	Lifetime of the packet
RHL	Shall always be smaller than or equal to the maximum hop limit (*MHL*) in the *Common Header* (clause 10.3.4) Source: 1 if parameter Packet transport type in the service primitive TRANSP_CORE.request is SHB, or if Header type HT = 1 (BEACON) Value of optional Maximum hop limit parameter from service primitive TRANSP_CORE.request Otherwise GN protocol constant `itsGnDefaultHopLimit` if TRANSP_CORE.request parameter Packet transport type is GUC, GBC, GBC or TSB Forwarder: Decrement *RHL* by one	Remaining hop limit

10.3.3 *Basic Header* processing

When a GAGH in a forwarder, receiver or destination GeoAdhoc router processes a *Basic Header*, the GAGH shall execute the following operations upon reception of a GeoNetworking packet:

1) check the *Version* field of the *Basic Header:*

   a) if the value of the *Version* field equals the GN protocol constant `itsGnProtocolVersion`, continue the execution of further steps;

NOTE 1: For other values of the *Version* field, the implementation may select the appropriate protocol decoder.

2) check the *NH* field of the *Basic Header* (table 5):

a) if NH = 0 (*ANY*) or NH = 1 (*Common Header*), proceed processing the *Common Header* as specified in clause 10.3.5;

b) if NH = 2 (*Secured Packet*):

i) execute the SN-DECAP service with the parameter setting in table 19;

Table 19: Parameter settings in the service primitive *SN-DECAP.request*

Parameter name	Parameter setting
sec_packet_length	Length of the *Secured Packet* [octets]
sec_packet	The *Secured Packet* to be verified

ii) process the service primitive *SN-DECAP.confirm;*

iii) if the parameter *report* of the service primitive *SN-ENCAP.confirm* indicates that the packet was correctly verified and decrypted (parameter *report* = SUCCESS):

1) process the parameters *plaintext_packet*, *certificate_id*, *its_aid_length*, *its_aid*, *permissions length* and *permissions* carried in the *SN-DECAP.confirm* parameter and proceed processing the *Common Header* as specified in clause 10.3.5;

NOTE 2: In an implementation the *Secured Packet* should be kept, in order to forward the signed/encrypted packet without additional security processing.

iv) otherwise (parameter *report* != SUCCESS):

1) if the GN protocol constant `itsGnSnDecapResultHandling` is set to STRICT (0) discard the packet and omit the execution of further steps;

2) if the GN protocol constant `itsGnSnDecapResultHandling` is set to NON-STRICT (1) pass the payload of the GN-PDU to the upper protocol entity by means of a service primitive *TRANSP_CORE.indication*.

NOTE 3: The purpose for passing the GN-PDU to the upper protocol entity with incorrect result of verification and decryption may improve security assessment of messages at the Facilities layer. Details are implementation specific.

10.3.4 *Common Header* field settings

For all GeoNetworking packets, the fields of the *Common Header* shall be set as specified in table 20.

Table 20: Field settings for the *Common Header*

Field name	Field setting	Description
NH	0 (ANY) if *Header type* HT = 1 (BEACON) 1 (BTP-A) if *TRANSP_CORE.request* parameter *Upper protocol entity* is BTP-A 2 (BTP-B) if *TRANSP_CORE.request* parameter *Upper protocol entity* is BTP-B 3 (IPv6) if *TRANSP_CORE.request* parameter *Upper protocol entity* is Ipv6	Next header (table 8 in clause 9.7.3)
Reserved	Set to 0	Reserved
HT	1 (BEACON) if the GeoNetworking packet is a Beacon 2 (GEOUNICAST) if *TRANSP_CORE.request* parameter *Packet transport type* is GeoUnicast 3 (GEOANYCAST) if *TRANSP_CORE.request* parameter *Packet transport type* is GeoAnycast 4 (GEOBROADCAST) if *TRANSP_CORE.request* parameter *Packet transport type* is GBC 5 (TSB) if *TRANSP_CORE.request* parameter *Packet transport type* is TSB 5 (TSB) if *TRANSP_CORE.request* parameter *Packet transport type* is SHB 6 (LS) if the GeoNetworking packet is a LS Request or a LS Reply packet	Header type (table 9 in clause 9.7.4)
HST	As specified in table 9 in clause 9.7.4	Header sub-type
TC	*Traffic class* defined in service primitive *TRANSP_CORE.request* parameter *Traffic class* or GN protocol constant `itsGnDefaultTrafficClass` if the service primitive *TRANSP_CORE.request* parameter *Traffic class* is not available	Traffic class encoding specified in clause 9.7.5
Flags	Bit 0: GN protocol constant `itsGnIsMobile` Bit 1 to 7: Reserved; set to 0	Bit 0: Indicates whether the ITS-S is mobile or stationary
PL	• 0 for Beacon, LS Request and LS Reply packets • Size of T/GN6-SDU defined in service primitive *TRANSP_CORE.request* otherwise	Payload length in octets
MHL	• Source: • 1 if *TRANSP_CORE.request* parameter *Packet transport type* is SHB or GeoNetworking packet is Beacon • Value of optional `Maximum hop limit` parameter from service primitive *TRANSP_CORE.request* • GN protocol constant `itsGnDefaultHopLimit` if *TRANSP_CORE.request* parameter *Packet transport type* is GUC, GAC, GBC or TSB • GN protocol constant `itsGnDefaultHopLimit` for LS Request and LS Reply packets	Maximum hop limit
Reserved	Set to 0	Reserved
NOTE: TSB and SHB carry the same value in the *HT* field (equals 5), but have a different value in the *HST* field, i.e. HST = 0 for SHB and HST = 1 for TSB (table 9 in clause 9.7.4).		

10.3.5 *Common Header* processing

When a GAGH (forwarder, receiver, destination) processes a *Common Header*, the GAGH shall execute the following operations upon reception of a GeoNetworking packet:

1) check the *MHL* field of the *Common Header:*
   a) compare *MHL* with the value of the *RHL* field of the *Basic Header*; if *MHL* < *RHL* discard the packet and omit the execution of further steps;
2) process the *BC forwarding packet buffer*:
   a) if the *BC forwarding packet buffer* (clause 8.6) is not empty, forward the stored packets and remove them from the *BC forwarding packet buffer*;

NOTE 1: The forwarding algorithm having caused the buffering needs to be re-executed.

3) check the *HT* field of the *Common Header:*
   a) if HT = 0 (ANY) discard the packet and omit the execution of further steps;
   b) if HT = 1 (BEACON) execute the steps specified in clause 10.3.6;
   c) if HT = 2 (GEOUNICAST) execute the steps specified in clause 10.3.8.3 and clause 10.3.8.4;
   d) if HT = 3 (GEOANYCAST) execute the steps specified in clause 10.3.12.3;
   e) if HT = 4 (GEOBROADCAST) execute the steps specified in clause 10.3.11.3;
   f) if HT = 5 (TSB) execute the steps specified in clause 10.3.9.3 (HST = MULTI_HOP) and clause 10.3.10.3 (HST = SINGLE_HOP);
   g) if HT = 6 (LS) execute the steps specified in clause 10.3.7.2 and clause 10.3.7.3.

NOTE 2: In 3a) to 3g) only the steps after "*Common Header* processing" need to be executed in the corresponding clauses defining the packet handling procedures.

10.3.6 Beacon packet handling

10.3.6.1 General

Beaconing is used to periodically advertise a GeoAdhoc router's position vector to its neighbours.

A BEACON packet shall be sent periodically by each active GAGH unless the GAGH sends another GeoNetworking packet that carries the EPV.

NOTE: In one possible implementation a timer is reset upon transmission of a SHB packet.

10.3.6.2 Source operations

Upon activation, a GAGH shall execute the following operations:

1) create a GN-PDU with a Beacon packet header (clause 9.8.6):
   a) set the fields of the *Basic Header* (clause 10.3.2);
   b) set the fields of the *Common Header* (clause 10.3.4);
   c) set the fields of the Beacon *Extended Header* (table 21);

Table 21: Field settings for the Beacon *Extended Header*

Field name	Field setting	Description
SO PV	Actual values of the EPV as specified in clause 8.3	PV of the ego GeoAdhoc router (source of the GeoNetworking packet)

2) if the GN protocol constant `itsGnSecurity` is set to ENABLED:

a) send a service primitive *SN-ENCAP.request* with the parameter setting in table 22;

Table 22: Parameter settings in the service primitive SN-ENCAP.request for a BEACON packet

Parameter name	Parameter setting
tbe_packet_length	Length of the Beacon
tbe_packet	*Common header* + Beacon Extended header to be signed
sec_profile	If the GN protocol constant `itsGnSecurity` is set to ENABLED, the value of the parameter *sec_profile* is set to a default security profile. The specification of the default security profile is out of scope of the present document.
Its_aid _length	2 (see note 1)
its_aid	141 = 0x8d (see note 1)
permissions_length	0 (see note 2)
permissions	Void
context_information	0
target_id_list_length	0
target_id_list	Void
NOTE 1: See ETSI TS 102 965 [i.6]. NOTE 2: The *permissions_length* parameter being set to zero indicates that the SSP is null and absent.	

b) process the service primitive *SN-ENCAP.confirm* and append the *Secured Packet* carried by the *sec_packet* parameter of the service primitive *SN-ENCAP.confirm* to the *Basic Header*;

3) execute media-dependent procedures; if the GN protocol constant `itsGnIfType` is set to:

a) UNSPECIFIED then no media-dependent procedures are specified;

b) for other values of `itsGnIfType` media dependent procedures may be defined elsewhere;

4) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the Broadcast address of the ALI;

5) initialize the timer for the periodic transmission of beacons T_{beacon} with a timeout set to (`itsGnBeaconServiceRetransmitTimer` + RAND[0,`itsGnBeaconServiceMaxJitter`]), whereas `itsGnBeaconServiceRetransmitTimer` and `itsGnBeaconServiceMaxJitter` represent GN protocol constant values.

NOTE 1: The RAND function introduces a random component for the timer to avoid synchronization issues among GeoAdhoc routers.

If the timer T_{beacon} expires, the source shall execute the following operations:

1) create a GN-PDU with a BEACON packet header (clause 9.8.6):

a) set the fields of the *Basic Header* to the values specified in clause 10.3.2;

b) set the fields of the *Common Header* to the values specified in clause 10.3.4;

c) set the fields of the Beacon *Extended Header* as specified in table 21;

2) if the GN protocol constant `itsGnSecurity` is set to ENABLED:

a) send a service primitive *SN-ENCAP.request* with the parameter setting in table 22;

b) process the service primitive *SN-ENCAP.confirm* and append the *Secured Packet* carried by the *sec_packet* parameter of the service primitive *SN-ENCAP.confirm* to the *Basic Header*;

3) execute media-dependent procedures; if the GN protocol constant `itsGnIfType` is set to:

a) UNSPECIFIED then no media-dependent procedures are specified;

b) for other values of `itsGnIfType` media dependent procedures may be defined elsewhere;

4) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the Broadcast address of the ALI;

5) set the timer T_{beacon} to a timeout value (`itsGnBeaconServiceRetransmitTimer +` RAND[0,`itsGnBeaconServiceMaxJitter`]).

NOTE 2: The GAGH resets the timer T_{beacon} for every sent GeoNetworking packet that carries a LPV.

10.3.6.3 Receiver operations

Receiver operations of Beacon packets are identical to the handling procedures of the SHB packet (clause 10.3.10.3) except step 8 (pass the payload of the GN-PDU to the upper protocol entity).

10.3.7 Location service packet handling

10.3.7.1 Source operations

10.3.7.1.1 Overview

Three cases are distinguished for the source operations:

1) source operation for initial LS Request (clause 10.3.7.1.2);

2) source operation for LS Request re-transmission (clause 10.3.7.1.3);

3) source operation for LS Reply (clause 10.3.7.1.4).

10.3.7.1.2 Operation for initial LS Request

When a source has a T/GN6-SDU to send and has no position vector information for the destination address, the source shall invoke the location service and shall execute the following operations:

1) check whether a LS for the sought *GN_ADDR* is in progress, i.e. the flag *LS_pending* is set TRUE:

   a) if *LS_pending* is TRUE for the sought GN_ADDR, the packet shall be buffered in the *LS packet buffer* (clause 8.5) and the execution of the next steps shall be omitted;

2) create a GN-PDU with the T/GN6-SDU as payload and a TSB packet header (clause 9.8.3):

   a) set the fields of the *Basic Header* to the values specified in clause 10.3.2;

   b) set the fields of the *Common Header* to the values specified in clause 10.3.4;

   c) set the fields of the LS Request *Extended Header* to the values specified in table 23;

Table 23: Field settings for the LS Request *Extended Header*

Field name	Field setting	Description
SN	Actual value of the local sequence number as specified in clause 8.4	Sequence number of the packet
SO PV	Actual values of the EPV as specified in clause 8.3	Position vector containing the reference position of the ego GeoAdhoc router (source of the GeoNetworking packet)
Request GN_ADDR	GN_ADDR from *TRANSP_CORE.request* parameter *Destination* (clause J.2)	GeoNetworking address of the sought GeoAdhoc router

3) if the GN protocol constant `itsGnSecurity` is set to ENABLED:

   a) send a service primitive *SN-ENCAP.request* with the parameter setting in table 24;

Table 24: Parameter settings in the service primitive SN-ENCAP.request for a LS Request packet

Parameter name	Parameter setting (in case of security as specified in note 2)
tbe_packet_length	Length of the LS Request
tbe_packet	*Common header* + *LS Request Extended header* to be signed
sec_profile	If the GN protocol constant `itsGnSecurity` is set to ENABLED, the value of the parameter *sec_profile* is set to a default security profile. The specification of the default security profile is out of scope of the present document
its_aid _length	2 (see note 1)
its_aid	141 = 0x8d (see note 1)
permissions_length	0 (see note 2)
permissions	Void
context_information	0
target_id_list_length	0
target_id_list	Void
NOTE 1: See ETSI TS 102 965 [i.6]. NOTE 2: The *permissions_length* parameter being set to zero indicates that the SSP is null and absent.	

b) process the service primitive *SN-ENCAP.confirm* and append the *Secured Packet* carried by the *sec_packet* parameter of the service primitive *SN-ENCAP.confirm* to the *Basic Header*;

4) execute media-dependent procedures; if the *Communication profile* parameter of the service primitive *TRANSP_CORE.request* is set to:

a) UNSPECIFIED then no media-dependent procedures are specified;

b) for other values media dependent procedures may be defined elsewhere;

5) start a timer $T_{LS,\ GN\_ADDR}$ with a timeout set to the value of the GN protocol constant `itsGnLocationServiceRetransmitTimer`;

6) initialize the LS retransmit counter for the GeoAdhoc router *GN_ADDR* $RTC_{LS,\ GN\_ADDR}$ to 0;

7) add a LocTE for the sought *GN_ADDR* in the LocT and set the flag *LS_pending* to TRUE;

8) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the Broadcast address of the ALI.

10.3.7.1.3 Operation for LS Request re-transmission

If the timer $T_{LS,\ GN\_ADDR}$ for the *GN_ADDR* expires, the source shall execute the following operations:

1) check the retransmit counter $RTC_{LS,\ GN\_ADDR}$:

a) if the retransmit counter is less than the maximum number of LS retransmissions set by the GN protocol constant `itsGnLocationServiceMaxRetrans`,
i.e. $RTC_{LS,\ GN\_ADDR}$ < `itsGnLocationServiceMaxRetrans,` the GAGH shall:

i) re- issue a LS Request packet with the format as specified in clause 9.8.7 as a TSB packet and the field setting for the LS Request *Extended Header* as specified in clause 10.3.7.1.2, table 23;

ii) restart the timer $T_{LS,\ GN\_ADDR}$ with a timeout set to the value of the GN protocol constant `itsGnLocationServiceRetransmitTimer`; and

iii) increment the retransmit counter $RTC_{LS,\ GN\_ADDR}$;

b) if the retransmit counter is greater than or equals the maximum number of LS retransmissions set by the GN protocol constant `itsGnLocationServiceMaxRetrans`,
i.e. $RTC_{LS,\ GN\_ADDR}$ ≥ `itsGnLocationServiceMaxRetrans,` the GAGH shall:

i) remove the pending packets for the sought GN_ADDR from the *LS packet buffer* (clause 8.5);

ii) remove the LocTE for the sought *GN_ADDR*.

10.3.7.1.4 Operation for LS Reply

If the source receives a LS Reply packet for the sought *GN_ADDR*, the source shall execute the following operations:

1) *Basic Header* processing (clause 10.3.3);

2) *Common Header* processing (clause 10.3.5);

3) execute DPD as specified in clause A.2; if the LS Reply packet is a duplicate, discard the packet and omit the execution of further steps;

4) update *PV(SO)* in the LocT with the *SO PV* of the LS Reply *Extended Header* (clause C.2);

5) update *PDR(SO)* in the LocT (clause B.2);

6) flush the *LS packet buffer* (clause 8.5) for the sought GN_ADDR and forward the stored packets;

7) flush packet buffers (*SO LS packet buffer, SO UC forwarding packet buffer*):

   a) if SO LS_pending is TRUE:

      i) forward the packets in the SO *LS packet buffer* and remove them from the buffer (clause 8.5);

      ii) set *SO LS_pending* to false;

   b) if the *UC forwarding packet buffer* (clause 8.6) for *SO* is not empty, forward the stored packets and remove them from the *UC forwarding packet buffer*;

8) set the flag LS_pending for the sought GN_ADDR to false;

9) stop the timer $T_{LS,\ GN\_ADDR}$;

10) reset the re-transmit counter $RTC_{LS,\ GN\_ADDR}$.

10.3.7.2 Forwarder operations

If a GAGH receives a LS Request packet and the *Request GN_ADDR* field in the LS Request header does not match its *GN_ADDR*, the GAGH shall handle the packet according to the packet handling procedure for TSB (clause 10.3.9.3), except step 7 for passing the payload of the GN-PDU to the upper protocol entity.

If a GAGH receives a LS Reply packet and the *GN_ADDR* in the *DE PV* of the LS Reply packet does not match its *GN_ADDR*, the GAGH shall handle the packet according to the packet handling operations (forwarder) for GUC (clause 10.3.8.3).

NOTE: The *Basic Header* and *Common Header* processing are part of the GeoUnicast and TSB packet handling procedure, respectively.

10.3.7.3 Destination operations

On reception of a LS Request packet, the GAGH shall check the *Request GN_ADDR* field. If this MID field matches the MID field of its GN_ADDR, the GAGH shall execute the following operations:

1) *Basic Header* processing (clause 10.3.3);

2) *Common Header* processing (clause 10.3.5);

3) execute DPD as specified in clause A.2; if the LS Request packet is a duplicate, discard the packet and omit the execution of further steps;

4) execute DAD as specified in clause 10.2.1.5;

5) if the LocTE(SO) does not exist:

   a) create *PV(SO)* in the LocT with the *SO PV* fields of the LS Request *Extended Header* (clause C.2);

   b) set the *IS_NEIGHBOUR* flag of the *SO* LocTE to FALSE;

   c) set *PDR(SO)* in the SO LocTE (clause B.2);

6) if the LocTE(SO) exists:

   a) update *PV(SO)* in the LocT with the *SO PV* fields of the LS Request *Extended Header* (clause C.2);

   b) update *PDR(SO)* in the SO LocTE (clause B.2);

NOTE: The *IS_NEIGHBOUR* flag of the SO LocTE remains unchanged.

7) if the return value of the forwarding algorithm is 0 (packet is buffered in the *UC forwarding packet buffer*) or -1 (packet is discarded), omit the execution of further steps;

8) create a GN-PDU with a GUC packet header (clause 9.8.2):

   a) set the fields of the *Basic Header* (clause 10.3.2);

   b) set the fields of the *Common Header* (clause 10.3.4);

   c) set the fields of the GUC *Extended Header* (table 25);

Table 25: Field settings for the GUC *Extended Header* in LS

Field name	Field setting	Description
SN	Actual value of the local sequence number (clause 8.4)	Sequence number of the packet
SO PV	Actual values of the EPV (clause 8.3)	Position vector containing the reference position of the ego GeoAdhoc router (source of the GeoNetworking packet)
DE PV	Actual values of the LocTE for the destination	Position vector containing the reference position of the destination

9) execute the forwarding algorithm (annex E):

   a) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 0 (UNSPECIFIED), execute the GF algorithm as specified in clause E.2;

   b) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 1 (GREEDY), execute the GF algorithm as specified in clause E.2;

   c) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 2 (CBF), set the ALI address to the Broadcast ALI address;

10) if the optional *Security profile* parameter in the service primitive *TRANSP_CORE.request* is set:

   a) send a service primitive *SN-ENCAP.request* with the parameter setting in table 26;

Table 26: Parameter settings in the service primitive SN-ENCAP.request for the GUC packet in LS

Parameter name	Parameter setting
tbe_packet_length	Length of the GUC header + BTP header + payload
tbe_packet	*Common header* + GUC header + BTP header + payload to be signed
sec_profile	The value of the parameter *Security profile* in the service primitive *TRANSP_CORE.request* (see note 1)
its_aid _length	2 (see note 2)
its_aid	141 = 0x8d (see note 2)
permissions_length	0 (see note 3)
permissions	Void
context_information	0
target_id_list_length	Void
target_id_list	0
NOTE 1: If the parameter *Security profile* in the service primitive *TRANSP_CORE.request* is not set and the GN protocol constant `itsGnSecurity` is set to ENABLED, a default security profile is used. The specification of the default security profile is out of scope of the present document. NOTE 2: See ETSI TS 102 965 [i.6]. NOTE 3: The *permissions_length* parameter being set to zero indicates that the SSP is null and absent.	

b) process the service primitive *SN-ENCAP.confirm* and append the *Secured Packet* carried by the *sec_packet* parameter of the service primitive *SN-ENCAP.confirm* to the *Basic Header*;

11) execute media-dependent procedures; if the *Communication profile* parameter of the service primitive *TRANSP_CORE.request* is set to:

a) UNSPECIFIED then no media-dependent procedures are specified;

b) for other values media dependent procedures may be defined elsewhere;

12) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the link layer address of the next hop *LL_ADDR_NH*.

10.3.8 GUC packet handling

10.3.8.1 General

This clause specifies the operations of a GAGH to handle a GUC packet. The following clauses define the operations of the source, forwarder and destination.

GeoUnicast forwarding applies the algorithm selected by the setting of the value in the GN protocol constant `itsGnNonAreaForwardingAlgorithm` and specified in annex E.

10.3.8.2 Source operations

On reception of a service primitive *TRANSP_CORE.request* with a *Packet transport type* parameter set to *GeoUnicast*, the source shall execute the following operations:

1) create a GN-PDU with the T/GN6-SDU as payload and a GUC packet header (clause 9.8.2):

a) set the fields of the *Basic Header* (clause 10.3.2);

b) set the fields of the *Common Header* (clause 10.3.4);

c) set the fields of the GUC *Extended Header* (table 27);

Table 27: Field settings for the GUC *Extended Header*

Field name	Field setting	Description
SN	Actual value of the local sequence number (clause 8.4)	Sequence number of the packet
SO PV	Actual values of the EPV (clause 8.3)	Position vector containing the reference position of the ego GeoAdhoc router (source of the GeoNetworking packet)
DE PV	Actual values of the LocTE for the destination	Position vector containing the reference position of the destination or empty in case the destination position is not yet available

2) check whether the entry of the position vector for DE in its LocT is valid:

   a) if no valid position vector information is available, the source shall invoke the location service as specified in clause 10.3.7.1.2 and omit the execution of further steps. Otherwise, the source shall proceed with step 2;

3) if no neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and *SCF* for the traffic class in the service primitive *TRANSP_CORE.request* parameter *Traffic class* is enabled, then buffer the GUC packet in the *UC forwarding packet buffer* and omit the execution of further steps;

4) execute the forwarding algorithm (see annex E):

   a) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 0 (UNSPECIFIED), execute the GF algorithm as specified in clause E.2;

   b) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 1 (GREEDY), execute the GF algorithm as specified in clause E.2;

   c) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 2 (CBF), set the LL_ADDR_NH address to the Broadcast link layer address;

5) if the return value of the forwarding algorithm is 0 (packet is buffered in the *UC forwarding packet buffer*) or -1 (packet is discarded), omit the execution of further steps;

6) if the optional *Security profile* parameter in the service primitive *TRANSP_CORE.request* is set:

   a) send a service primitive *SN-ENCAP.request* with the parameter setting in table 28;

Table 28: Parameter settings in the service primitive SN-ENCAP.request for the GUC packet

Parameter name	Parameter setting
tbe_packet_length	Length of the GUC header + BTP header + payload
tbe_packet	*Common header* + GUC header + BTP header + payload to be signed
sec_profile	Value of the corresponding parameter in the service primitive *TRANSP_CORE.request* (clause J.2) (optional)
its_aid_length	
its_aid	
permissions_length	
permissions	
context_information	
target_id_list_length	
target_id_list	
NOTE: If the security-related parameters in the service primitive *TRANSP_CORE.request* are not set and the GN protocol constant `itsGnSecurity` is set to ENABLED, a default security profile is used. The specification of the default security profile is out of scope of the present document.	

   b) process the service primitive *SN-ENCAP.confirm* and append the *Secured Packet* carried by the *sec_packet* parameter of the service primitive *SN-ENCAP.confirm* to the *Basic Header*;

7) if the optional *Repetition interval* parameter in the *TRANSP_CORE.request* parameter is set:

   a) save the GUC packet;

   b) retransmit the packet with a period as specified in *Repetition interval* until the maximum repetition time of the packet is expired;

NOTE 1: The maximum repetition time of the packet is specified in the *Maximum repetition time* parameter of the service primitive *TRANSP_CORE.request*.

NOTE 2: For every retransmission, the source operations need to be re-executed.

NOTE 3: The functionality of packet repetition is optional.

8) execute media-dependent procedures; if the *Communication profile* parameter of the service primitive *TRANSP_CORE.request* is set to:

   a) UNSPECIFIED then no media-dependent procedures are specified;

   b) for other values media dependent procedures may be defined elsewhere;

9) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the link layer address of the next hop *LL_ADDR_NH*.

10.3.8.3 Forwarder operations

On reception of a GUC packet, a GAGH shall check the *GN_ADDR* field in the *DE PV* of the GUC packet header. If this address does not match its *GN_ADDR*, the GAGH shall execute the following operations:

1) *Basic Header* processing (clause 10.3.3);

2) *Common Header* processing (clause 10.3.5);

3) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 0 (UNSPECIFIED) or to 1 (GREEDY), execute DPD as specified in clause A.2; if the GUC packet is a duplicate, discard the packet and omit the execution of further steps;

NOTE 1: For CBF (`itsGnNonAreaForwardingAlgorithm` is set to 2), the algorithm relies on the processing of duplicate packets and their handling is part of the forwarding algorithm.

4) execute DAD as specified in clause 10.2.1.5;

5) if the LocTE(SO) does not exist:

   a) create *PV(SO)* in the LocT with the *SO PV* fields of the GUC *Extended Header* (clause C.2);

   b) set the *IS_NEIGHBOUR* flag of the *SO* LocTE to FALSE;

   c) update *PDR(SO)* in the LocT (clause B.2);

6) if the LocTE(SO) exists:

   a) update *PV(SO)* in the LocT with the *SO PV* fields of the GUC *Extended Header* (clause C.2);

   b) update *PDR(SO)* in the LocT (clause B.2);

NOTE 2: The *IS_NEIGHBOUR* flag of the SO LocTE remains unchanged.

7) if the DE LocTE does not exist or the *IS_NEIGHBOUR* flag of the existing DE LocTE is not set:

   a) if the *NH* field of the *Basic Header* NH = 0 (ANY) or NH = 1 (*Common Header*), update the PV(DE) in the LocT with the DE PV value of the GUC packet (clause C.2);

NOTE 3: The DE PV is a *Short Position Vector* (SPV) and does not carry all fields required to set the PV(DE) in the LocT, i.e. no PAI, speed, heading. Therefore, if the PV in the LocT is updated, the fields PAI, speed, heading are set to 0.

NOTE 4: If the *IS_NEIGHBOUR* flag of the DE LocTE is set, the DE LocTE is not updated.

8) if the DE LocTE exists and the *IS_NEIGHBOUR* flag of the existing DE LocTE is set:

   a) if the *NH* field of the *Basic Header* NH = 0 (ANY) or NH = 1 (*Common Header*), update the DE PV field in the GUC packet with the PV(DE) in the LocT (clause C.3);

NOTE 5: If the DE PV overwrites an existing LocTE with an *IS_NEIGHBOUR* flag set, this entry would not be considered in the forwarding algorithm due to the PAI set to 0.

NOTE 6: The *DE PV* fields are not updated if NH = 2 (*Secured Packet*).

9) flush packet buffers (*SO LS packet buffer*, *SO UC forwarding packet buffer*):

   a) if LS_pending(SO) is TRUE:

      i) forward the stored packets and remove them from the *SO LS packet buffer* (clause 8.5);

      ii) set *LS_pending(SO)* to false;

   b) if the *UC forwarding packet buffer* (clause 8.6) for *SO* is not empty, flush the *UC forwarding packet buffer* and forward the stored packets;

10) decrement the *RHL* value:

   a) if *RHL* = 0 discard the packet and omit the execution of further steps;

   b) if *RHL* > 0 update the field of the *Basic Header*, i.e. the *RHL* field with the decremented *RHL* value;

11) if no neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and SCF for the traffic class in the *TC* field of the *Common Header* is set, buffer the GUC packet in the *UC forwarding packet buffer* and omit the execution of further steps;

12) execute the forwarding algorithm (annex E):

   a) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 0 (UNSPECIFIED), execute the GF algorithm as specified in clause E.2;

   b) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 1 (GREEDY), execute the GF algorithm as specified in clause E.2;

   c) if the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 2 (CBF), execute the CBF algorithm as specified in clause E.3;

13) if the return value of the forwarding algorithm is 0 (packet is buffered in a forwarding packet buffer) or -1 (packet is discarded), omit the execution of further steps;

14) execute media-dependent procedures; if the GN protocol constant `itsGnIfType` is set to:

   a) UNSPECIFIED then no media-dependent procedures are specified;

   b) for other values of `itsGnIfType` media dependent procedures may be defined elsewhere;

15) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the link layer address of the next hop LL_ADDR_NH.

10.3.8.4 Destination operations

On reception of a GUC packet, the GeoAdhoc router shall check the *GN_ADDR* field in the *DE PV* of the GUC packet header. If this address matches its *GN_ADDR*, the GeoAdhoc router shall execute the following operations:

1) *Basic Header* processing (clause 10.3.3);

2) *Common Header* processing (clause 10.3.5);

3) execute DPD as specified in clause A.2; if the GUC packet is a duplicate, discard the packet and omit the execution of further steps;

4) execute DAD as specified in clause 10.2.1.5;

5) if the *LocTE(SO)* does not exist:

   a) create *PV(SO)* in the LocT with the *SO PV* fields of the GUC *Extended Header* (clause C.2);

   b) set the *IS_NEIGHBOUR* flag of the *SO* LocTE to FALSE;

   c) set the *PDR(SO)* in *SO* LocT (clause B.2);

6) if the LocTE(SO) exists:

   a) update *PV(SO)* in the LocT with the *SO PV* fields of the GUC *Extended Header* (clause C.2);

   b) update the *PDR(SO)* in the LocT (clause B.2);

NOTE: The *IS_NEIGHBOUR* flag of the *SO* LocTE remains unchanged.

7) flush packet buffers (*SO LS packet buffer*, *SO UC forwarding packet buffer*):

   a) if LS_pending(SO) is TRUE:

      i) forward the stored packets and remove them from the *SO LS packet buffer* (clause 8.5);

      ii) set *LS_pending(SO)* to false;

   b) if the *UC forwarding packet buffer* (clause 8.6) for *SO* is not empty, flush the *UC forwarding packet buffer* and forward the stored packets;

8) pass the payload of the GN-PDU to the upper protocol entity by means of a service primitive *TRANSP_CORE.indication* with the parameter settings in table 29.

Table 29: Parameter settings in the service primitive *TRANSP_CORE.indication* to indicate a received GUC packet

Parameter name	Parameter setting
Upper protocol entity	BTP if NH = 1 (BTP-A) BTP if NH = 2 (BTP-B) Ipv6 if NH = 3 (Ipv6) (NH encoding see table 8 in clause 9.7.3)
Packet transport type	GeoUnicast
Destination	*DE GN_ADDR*
Source position vector	Values of *SO PV* from *Extended Header*
Security report	Value of the corresponding security-related parameter in the service primitive *SN-DECAP.indication*
Certificate id	
ITS-AID length	
ITS-AID	
Security permissions length	
Security permissions	
Traffic class	Value of *TC* field from *Common Header*
Remaining packet lifetime	Value of *LT* from *Basic Header*
Remaining hop limit	Value of *RHL* from *Basic Header*
Length	Length of the GN-PDU payload
Data	GN-PDU payload

10.3.9 TSB packet handling

10.3.9.1 General

This clause specifies the operations of a GAGH to handle a TSB packet. The following clauses define the operations of the source and forwarder/receiver.

NOTE: In TSB, a forwarder is also always a receiver. Therefore, the roles are not distinguished.

10.3.9.2 Source operations

On reception of a service primitive *TRANSP_CORE.request* with a *Packet transport type* parameter set to *TSB,* the source shall execute the following operations:

1) create a GN-PDU with the T/GN6-SDU as payload and a TSB packet header (clause 9.8.3):

   a) set the fields of the *Basic Header* (clause 10.3.2);

   b) set the fields of the *Common Header* (clause 10.3.4);

   c) set the fields of the TSB *Extended Header* (table 30);

Table 30: Field settings for the TSB *Extended Header*

Field name	Field setting	Description
SN	Actual value of the local sequence number (clause 8.4)	Sequence number of the packet
Reserved	Set to 0 if not used for media-dependent operations	Reserved for media-dependent operations
SO PV	Actual values of the EPV (clause 8.3)	Position vector containing the reference position of the ego GeoAdhoc router (source of the GeoNetworking packet)

2) if the optional Security profile parameter in the service primitive *TRANSP_CORE.request* is set:

   a) send a service primitive *SN-ENCAP.request* as specified with the parameter setting in table 31;

Table 31: Parameter settings in the service primitive SN-ENCAP.request for the TSB packet

Parameter name	Parameter setting
tbe_packet_length	Length of the TSB header + BTP header + payload
tbe_packet	*Common header* + TSB header + BTP header + payload to be signed
sec_profile	The value of the corresponding parameter in the service primitive *TRANSP_CORE.request* (clause J.2)
its_aid_length	
its_aid	
permissions length	
permissions	
context_information	
target_id_list_length	
target_id_list	
NOTE: If the security-related parameters in the service primitive *TRANSP_CORE.request* are not set and the GN protocol constant `itsGnSecurity` is set to ENABLED, a default security profile is used. The specification of the default security profile is out of scope of the present document.	

   b) process the service primitive *SN-ENCAP.confirm* and append the *Secured Packet* carried by the *sec_packet* parameter of the service primitive *SN-ENCAP.confirm* to the *Basic Header*;

3) if no suitable neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and *SCF* for the traffic class in the service primitive *TRANSP_CORE.request* parameter *Traffic class* is enabled, then buffer the TSB packet in the *BC forwarding packet buffer* and omit the execution of further steps;

NOTE 1: If *SCF* for the traffic class is disabled, the TSB packet is never buffered but sent immediately with a broadcast link layer destination address.

NOTE 2: Buffered packets may be further processed when the GAGH receives a packet (e.g. SHB, GBC, BEACON, etc.), see the corresponding clauses on forwarder and receiver operations.

4) if the optional *Repetition interval* parameter in the *TRANSP_CORE.request* parameter is set:

   a) save the TSB packet;

   b) retransmit the packet with period as specified in *Repetition interval* until the maximum repetition time of the packet is expired;

NOTE 3: The maximum repetition time of the packet is specified in the *Maximum repetition time* parameter of the service primitive *TRANSP_CORE.request*.

NOTE 4: For every retransmission, the source operations need to be re-executed.

NOTE 5: The functionality of packet repetition is optional.

5) execute media-dependent procedures; if the *Communication profile* parameter of the service primitive *TRANSP_CORE.request* is set to:

   a) UNSPECIFIED then no media-dependent procedures are specified;

   b) for other values media dependent procedures may be defined elsewhere;

6) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the Broadcast address of the ALI.

10.3.9.3 Forwarder and receiver operations

On reception of a TSB packet, the GAGH shall execute the following operations:

1) *Basic Header* processing (clause 10.3.3);

2) *Common Header* processing (clause 10.3.5);

3) execute DPD as specified in clause A.2; if the TSB packet is a duplicate, discard the packet and omit the execution of further steps;

4) execute DAD as specified in clause 10.2.1.5;

5) if the LocTE(SO) does not exist:

   a) create *PV(SO)* in the LocT with the *SO PV* fields of the TSB *Extended Header* (clause C.2);

   b) set the *IS_NEIGHBOUR* flag of the *SO* LocTE to FALSE;

   c) set *PDR(SO)* in the SO LocTE (clause B.2);

6) if the LocTE(SO) exists:

   a) update *PV(SO)* in the LocT with the *SO PV* fields of the TSB *Extended Header* (clause C.2);

   b) update *PDR(SO)* in the LocT (clause B.2);

NOTE 1: The *IS_NEIGHBOUR* flag of the SO LocTE remains unchanged.

7) pass the payload of the GN-PDU to the upper protocol entity by means of a service primitive *TRANSP_CORE.indication* with the parameter settings in table 32;

Table 32: Parameter settings in the service primitive *TRANSP_CORE.indication* to indicate a received TSB packet

Parameter name	Parameter setting
Upper protocol entity	BTP if NH = 1 (BTP-A) BTP if NH = 2 (BTP-B) Ipv6 if NH = 3 (Ipv6) (NH encoding see table 8 in clause 9.7.3)
Packet transport type	TSB
Source position vector	Values of *SO PV* from *Extended Header*
Security report	Set to the value of the corresponding security-related parameter in the service primitive *SN-DECAP.indication* (optional)
Certificate id	
ITS-AID length	
ITS-AID	
Security permissions length	
Security permissions	
Traffic class	Value of *TC* field from *Common Header*
Remaining packet lifetime	Value of *LT* from *Basic Header*
Remaining hop limit	Value of *RHL* from *Basic Header*
Length	Length of the GN-PDU payload
Data	GN-PDU payload

8) flush packet buffers (*SO LS packet buffer*, *SO UC forwarding packet buffer*):

a) if LS_pending(SO) is TRUE:

i) forward the stored packets and remove them from the *SO LS packet buffer* (clause 8.5);

ii) set *LS_pending(SO)* to false;

b) if the *UC forwarding packet buffer* (clause 8.6) for *SO* is not empty, flush the *UC forwarding packet buffer* and forward the stored packets;

9) decrement the *RHL* value:

a) if *RHL* = 0 discard the packet and omit the execution of further steps;

b) if *RHL* > 0 update the field of the *Basic Header*, i.e. the *RHL* field with the decremented *RHL* value;

10) if no suitable neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and *SCF* for the traffic class in the *TC* field of the *Common Header* is set:

a) buffer the TSB packet in the *BC forwarding packet buffer* and omit the execution of further steps;

NOTE 2: If *SCF* for the traffic class is disabled, the TSB packet is never buffered but sent immediately with a broadcast link layer destination address.

NOTE 3: Buffered packets may be further processed when the GAGH receives a packet (e.g. SHB, GBC, BEACON, etc.), see the corresponding clauses on forwarder and receiver operations.

11) execute media-dependent procedures; if the GN protocol constant `itsGnIfType` is set to:

a) UNSPECIFIED then no media-dependent procedures are specified;

b) for other values of `itsGnIfType` media dependent procedures may be defined elsewhere;

12) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the Broadcast address of the ALI.

10.3.10 SHB packet handling

10.3.10.1 General

This clause specifies the operations of a GAGH to handle a SHB packet. The following clauses define the operations of the source and receiver.

NOTE: SHB packets are not forwarded. Therefore, no forwarder operations are specified.

10.3.10.2 Source operations

On reception of a service primitive *TRANSP_CORE.request* with a *Packet transport type* parameter set to *SHB*, the source shall execute the following operations:

1) create a GN-PDU with the T/GN6-SDU as payload and a SHB packet header (clause 9.8.4):

   a) set the fields of the *Basic Header* (clause 10.3.2);

   b) set the fields of the *Common Header* (clause 10.3.4);

   c) set the fields of the SHB *Extended Header* (table 33);

Table 33: Field settings for the SHB *Extended Header*

Field name	Field setting	Description
SO PV	Actual values of the EPV as specified in clause 8.3	PV of the ego GeoAdhoc router (source of the GeoNetworking packet)
Reserved	Set to 0 if not used for media-dependent operations	Reserved for media-dependent operations

2) if the optional Security profile parameter in the service primitive *TRANSP_CORE.request* is set:

   a) send a service primitive *SN-ENCAP.request* with the parameter setting in table 34;

Table 34: Parameter settings in the service primitive SN-ENCAP.request for the SHB packet

Parameter name	Parameter setting
tbe_packet_length	Length of the SHB header + BTP header + payload
tbe_packet	*Common header* + SHB header + BTP header + payload to be signed
sec_profile	The value of the corresponding parameter in the service primitive *TRANSP_CORE.request* (clause J.2)
its_aid_length	
its_aid	
permissions length	
permissions	
context_information	
target_id_list_length	
target_id_list	
NOTE: If the security-related parameters in the service primitive *TRANSP_CORE.request* are not set and the GN protocol constant `itsGnSecurity` is set to ENABLED, a default security profile is used. The specification of the default security profile is out of scope of the present document.	

   b) process the service primitive *SN-ENCAP.confirm* and append the *Secured Packet* carried by the *sec_packet* parameter of the service primitive *SN-ENCAP.confirm* to the *Basic Header*;

3) if no suitable neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and *SCF* for the traffic class in the service primitive *TRANSP_CORE.request* parameter *Traffic class* is set:

   a) buffer the SHB packet in the *BC forwarding packet buffer* and omit the execution of further steps;

NOTE 1: If *SCF* for the traffic class is disabled, the SHB packet is never buffered but sent immediately with a broadcast link layer destination address.

NOTE 2: Buffered packets may be further processed when the GAGH receives a packet (e.g. SHB, GBC, BEACON, etc.), see the corresponding clauses on forwarder and receiver operations.

4) if the optional *Repetition interval* parameter in the *TRANSP_CORE.request* parameter is set:

   a) save the SHB packet;

   b) retransmit the packet with a period as specified in *Repetition interval* parameter until the maximum repetition time of the packet is expired;

NOTE 3: The maximum repetition time of the packet is specified in the *Maximum repetition time* parameter of the service primitive *TRANSP_CORE.request*.

NOTE 4: For every retransmission, the source operations need to be re-executed.

NOTE 5: The functionality of packet repetition is optional.

5) execute media-dependent procedures; if the *Communication profile* parameter of the service primitive *TRANSP_CORE.request* is set to:

   a) UNSPECIFIED then no media-dependent procedures are specified;

   b) for other values media dependent procedures may be defined elsewhere;

6) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the Broadcast address of the ALI;

7) reset the beacon timer T_{beacon} to prevent the dissemination of an unnecessary beacon packet.

10.3.10.3 Receiver operations

On reception of a SHB packet, the GAGH shall execute the following operations:

1) *Basic Header* processing (clause 10.3.3);

2) *Common Header* processing (clause 10.3.5);

3) execute DAD as specified in clause 10.2.1.5;

4) update the *PV* in the *SO* LocTE with the *SO PV* fields of the SHB *Extended Header* (clause C.2);

5) update the *PDR* in the *SO* LocTE (clause B.2);

6) set the *IS_NEIGHBOUR* flag of the *SO* LocTE to TRUE;

7) pass the payload of the GN-PDU to the upper protocol entity by means of a service primitive *TRANSP_CORE.indication* with the parameter settings in table 35;

Table 35: Parameter settings in the service primitive *TRANSP_CORE.indication* to indicate a received SHB packet

Parameter name	Parameter setting
Upper protocol entity	BTP if NH = 1 (BTP-A) BTP if NH = 2 (BTP-B) Ipv6 if NH = 3 (Ipv6) (NH encoding see table 8 in clause 9.7.3)
Packet transport type,	SHB
Source position vector	Values of *SO PV* from SHB *Common Header*
Security report	Value of the corresponding security-related parameter in the service primitive *SN-DECAP.indication* (optional)
Certificate id	
ITS-AID length	
ITS-AID	
Security permissions length	
Security permissions	
Traffic class	Value of *TC* field from *Common Header*
Remaining packet lifetime	Value of *LT* from *Basic Header*
Remaining hop limit	Value of *RHL* from *Basic Header*
Length	Length of the GN-PDU payload
Data	GN-PDU payload

8) flush packet buffers (*SO LS packet buffer* an *SO UC forwarding packet buffer*):

   a) if SO LS_pending is TRUE:

      i) forward the stored packets and remove them from the buffer;

      ii) set *SO LS_pending* to false;

   b) if the *UC forwarding packet buffer* (clause 8.6) for the *SO* is not empty, forward the stored packets and remove them from the *UC forwarding packet buffer.*

10.3.11 GBC packet handling

10.3.11.1 General

This clause specifies the operations of a GAGH to handle a GBC packet. The following clauses define the operations of the source and forwarder/receiver.

NOTE: In GBC, a forwarder inside the target area acts also always as a receiver. Therefore, the roles are not distinguished.

10.3.11.2 Source operations

On reception of a service primitive *TRANSP_CORE.request* with a *Packet transport type* parameter set to *GeoBroadcast*, the source shall execute the following operations:

1) create a GN-PDU with the T/GN6-SDU as payload and a GBC packet header (clause 9.8.5):

   a) set the fields of the *Basic Header* (clause 10.3.2);

   b) set the fields of the *Common Header* (clause 10.3.4);

   c) set the fields of the GBC *Extended Header* (table 36);

Table 36: Field settings for the GBC *Extended Header*

Field name	Field setting	Description
SN	Actual value of the local sequence number (clause 8.4)	Sequence number of the packet
Reserved	Set to 0	Reserved
SO PV	Actual values of the EPV (clause 8.3)	Position vector containing the reference position of the ego GeoAdhoc router (source of the GeoNetworking packet)
GeoAreaPos Latitude	GeoAreaPos Latitude from service primitive *TRANSP_CORE.request*	GeoArea Area specification according to ETSI TS 103 899 [1]
GeoAreaPos Longitude	GeoAreaPos Longitude from service primitive *TRANSP_CORE.request*	
Distance a	Distance a from service primitive *TRANSP_CORE.request*	
Distance b	Distance a from service primitive *TRANSP_CORE.request*	
Angle	Angle from service primitive *TRANSP_CORE.request*	
Reserved	Set to 0	Reserved

2) if no neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and *SCF* for the traffic class in the service primitive *TRANSP_CORE.request* parameter *Traffic class* is enabled, then buffer the GBC packet in the *BC forwarding packet buffer* and omit the execution of further steps;

3) execute the forwarding algorithm procedures (starting with annex D);

4) if the return value of the forwarding algorithm is 0 (packet is buffered in the *BC forwarding packet buffer* or in the *CBF buffer*) or -1 (packet is discarded), omit the execution of further steps;

5) if the optional Security profile parameter in the service primitive *TRANSP_CORE.request* is set:

   a) send a service primitive *SN-ENCAP.request* with the parameter setting in table 37;

Table 37: Parameter settings in the service primitive SN-ENCAP.request for the GBC packet

Parameter name	Parameter setting
tbe_packet_length	Length of the GBC header + BTP header + payload
tbe_packet	*Common header* + GBC header + BTP header + payload to be signed
sec_services	Value of the corresponding parameter in the service primitive *TRANSP_CORE.request* (clause J.2)
its_aid_length	
its_aid	
permissions length	
permissions	
context_information	
target_id_list_length	
target_id_list	
NOTE: If the security-related parameters in the service primitive *TRANSP_CORE.request* are not set and the GN protocol constant `itsGnSecurity` is set to ENABLED, a default security profile is used. The specification of the default security profile is out of scope of the present document.	

   b) process the service primitive *SN-ENCAP.confirm* and append the *Secured Packet* carried by the *sec_packet* parameter of the service primitive *SN-ENCAP.confirm* to the *Basic Header*;

6) if the optional *Repetition interval* parameter in the *TRANSP_CORE.request* parameter is set:

   a) save the GBC packet;

   b) retransmit the packet with period as specified in *Repetition interval* until the maximum repetition time of the packet is expired;

NOTE 1: The maximum repetition time of the packet is specified in the *Maximum repetition time* parameter of the service primitive *TRANSP_CORE.request*.

NOTE 2: For every retransmission, the source operations need to be re-executed.

NOTE 3: The functionality of packet repetition is optional.

7) execute media-dependent procedures; if the *Communication profile* parameter of the service primitive *TRANSP_CORE.request* is set to:

   a) UNSPECIFIED then no media-dependent procedures are specified;

   b) for other values media dependent procedures may be defined elsewhere;

8) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the link layer address of the next hop *LL_ADDR_NH*.

10.3.11.3 Forwarder and receiver operations

On reception of a GBC packet, the GAGH shall execute the following operations:

1) *Basic Header* processing (clause 10.3.3);

2) *Common Header* processing (clause 10.3.5);

3) determine function F(x,y) as specified in ETSI TS 103 899 [1] clause 5:

   a) if $F(x,y) < 0$ (GeoAdhoc router is outside the geographical area) and the GN protocol constant `itsGnNonAreaForwardingAlgorithm` is set to 0 (UNSPECIFIED) or to 1 (GREEDY), execute DPD as specified in clause A.2; if the GBC packet is a duplicate, discard the packet and omit the execution of further steps;

NOTE 1a: For CBF forwarding algorithm (`itsGnNonAreaForwardingAlgorithm` is set to 2 or 3), the algorithm relies on the processing of duplicate packets and their handling is part of the forwarding algorithm.

   b) if $F(x,y) \geq 0$ (GeoAdhoc router is inside or at the border of the geographical area) and the GN protocol constant `itsGnAreaForwardingAlgorithm` is set to 0 (UNSPECIFIED) or to 1 (SIMPLE), execute DPD as specified in clause A.2; if the GBC packet is a duplicate, discard the packet and omit the execution of further steps;

NOTE 1b: For CBF and the Advanced forwarding algorithm (`itsGnAreaForwardingAlgorithm` is set to 2 or 3), the algorithm relies on the processing of duplicate packets and their handling is part of the forwarding algorithm.

4) execute DAD as specified in clause 10.2.1.5;

5) if the LocTE(SO) does not exist:

   a) create *PV(SO)* in the LocT with the *SO PV* fields of the GBC *Extended Header* (clause C.2);

   b) set the *IS_NEIGHBOUR* flag of the *SO* LocTE to FALSE;

   c) set *PDR(SO)* in the LocTE (clause B.2);

6) if the LocTE(SO) exists:

   a) update *PV(SO)* in the LocT with the *SO PV* fields of the GBC *Extended Header* (clause C.2);

   b) update *PDR(SO)* in the LocT (clause B.2);

NOTE 2: The *IS_NEIGHBOUR* flag of the SO LocTE remains unchanged.

7) determine function F(x,y) as specified in ETSI TS 103 899 [1] clause 5:

   a) if $F(x,y) \geq 0$ (GeoAdhoc router is inside or at the border of the geographical area), pass the payload of the GN-PDU to the upper protocol entity by means of a service primitive *TRANSP_CORE.indication* with the parameter settings in table 38;

NOTE 3: If the GeoAdhoc router is outside the geographical area, the GN-PDU will not be passed to the upper protocol entity.

NOTE 4: When an ITS-S invokes the function F(x,y), it is recommended to transform the GNSS coordinates into Cartesian coordinates using a suitable method to avoid large rounding errors on low-precision floating point systems. Such a suitable method is the haversine formula, for instance.

Table 38: Parameter settings in the service primitive *TRANSP_CORE.indication* to indicate a received GBC packet

Parameter name	Parameter setting
Upper protocol entity	BTP if NH = 1 (BTP-A) BTP if NH = 2 (BTP-B) Ipv6 if NH = 3 (Ipv6) (NH encoding see table 8 in clause 9.7.3)
Packet transport type,	GeoBroadcast
Destination	GeoArea (GeoPos, distance a, distance b, angle)
Source position vector	Values of *SO PV* from *Extended Header*
Security report	Value of the corresponding security-related parameter in the service primitive *SN-DECAP.indication* (optional)
Certificate id	
ITS-AID length	
ITS-AID	
Security permissions length	
Security permissions	
Traffic class	Value of *TC* field from *Common Header*
Remaining packet lifetime	Value of *LT* from *Basic Header*
Remaining hop limit	Value of *RHL* from *Basic Header*
Length	Length of the GN-PDU payload
Data	GN-PDU payload

8) flush packet buffers (*SO LS packet buffer*, *SO UC forwarding packet buffer*):

   a) if LS_pending(SO) is TRUE:

      i) forward the stored packets and remove them from the *SO LS packet buffer* (clause 8.5);

      ii) set *LS_pending(SO)* to false;

   b) if the *UC forwarding packet buffer* (clause 8.6) for *SO* is not empty, flush the *UC forwarding packet buffer* and forward the stored packets;

9) decrement the *RHL* value:

   a) if *RHL* = 0 discard the packet and omit the execution of further steps;

   b) if *RHL* > 0 update the field of the *Basic Header*, i.e. the *RHL* field with the decremented *RHL* value;

10) if no neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and SCF for the traffic class in the *TC* field of the *Common Header* is set, buffer the GBC packet in the *BC forwarding packet buffer* and omit the execution of further steps;

11) execute the forwarding algorithm procedures (starting with annex D);

12) if the return value of the forwarding algorithm is 0 (packet is buffered in a forwarding packet buffer) or -1 (packet is discarded), omit the execution of further steps;

NOTE 5: If *SCF* for the traffic class is disabled, the GBC packet is never buffered but sent immediately, except for the CBF packet buffer.

NOTE 6: Buffered packets may be further processed when the GAGH receives a packet (e.g. SHB, GBC, BEACON, etc.), see the corresponding clauses on forwarder and receiver operations.

13) execute media-dependent procedures; if the GN protocol constant `itsGnIfType` is set to:

   a) UNSPECIFIED then no media-dependent procedures are specified;

b) for other values of `itsGnIfType` media dependent procedures may be defined elsewhere;

14) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the link layer address of the next hop LL_ADDR_NH.

10.3.12 GAC packet handling

10.3.12.1 General

This clause specifies the operations of a GAGH to handle a GAC packet. The following clauses define the operations of the source and forwarder/receiver.

The operations for GAC packet handling are similar to those for GBC packet.

10.3.12.2 Source operations

The operations of the source of a GAC packet are identical with the source of a GBC packet as specified in clause 10.3.11.2.

NOTE: The procedure ensures that the GAC packet is sent at least once when the GeoAdhoc router is inside or at the border of the geographical area.

10.3.12.3 Forwarder and receiver operations

On reception of a GAC packet, the GAGH shall execute the following operations:

1) *Basic Header* processing (clause 10.3.3);

2) *Common Header* processing (clause 10.3.5);

3) execute DPD as specified in clause A.2; if the GAC packet is a duplicate, discard the packet and omit the execution of further steps;

4) execute DAD as specified in clause 10.2.1.5;

5) if the LocTE(SO) does not exist:

   a) create *PV(SO)* in the LocT with the *SO PV* fields of the GAC *Extended Header* (clause C.2);

   b) set the *IS_NEIGHBOUR* flag of the *SO* LocTE to FALSE;

   c) set *PDR(SO)* in the LocT (clause B.2);

6) if the LocTE(SO) exists:

   a) update *PV(SO)* in the LocT with the *SO PV* fields of the GAC *Extended Header* (clause C.2);

   b) update *PDR(SO)* in the LocT (clause B.2);

NOTE 1: The *IS_NEIGHBOUR* flag of the SO LocTE remains unchanged.

7) determine function F(x,y) as specified in ETSI TS 103 899 [1] clause 5;

8) flush packet buffers (*SO LS packet buffer*, *SO UC forwarding packet buffer*):

   a) if LS_pending(SO) is TRUE:

      i) forward the stored packets and remove them from the *SO LS packet buffer* (clause 8.5);

      ii) set *LS_pending(SO)* to false;

   b) if the *UC forwarding packet buffer* (clause 8.6) for *SO* is not empty, flush the *UC forwarding packet buffer* and forward the stored packets;

9) if $F(x, y) \geq 0$ (GeoAdhoc router is inside or at the border of the geographical area):

a) pass the payload of the GN-PDU to the upper protocol entity by means of a service primitive *TRANSP_CORE.indication* with the parameter settings in table 39;

Table 39: Parameter settings in the service primitive *TRANSP_CORE.indication* to indicate a received GAC packet

Parameter name	Parameter setting
Upper protocol entity	BTP if NH = 1 (BTP-A) BTP if NH = 2 (BTP-B) Ipv6 if NH = 3 (Ipv6) (NH encoding see table 8 in clause 9.7.3)
Packet transport type,	GeoAnycast
Destination	GeoArea (GeoPos, distance a, distance b, angle)
Source position vector	Values of SO PV from *Extended Header*
Security report	Value of the corresponding security-related parameter in the service primitive *SN-DECAP.indication* (optional)
Certificate id	
ITS-AID length	
ITS-AID	
Security permissions length	
Security permissions	
Traffic class	Value of TC field from *Common Header*
Remaining packet lifetime	Value of LT from *Basic Header*
Remaining hop limit	Value of *RHL* from *Basic Header*
Length	Length of the GN-PDU payload
Data	GN-PDU payload

b) omit the execution of further steps;

10) if $F(x, y) < 0$ (GeoAdhoc router is outside the geographical area):

a) decrement the *RHL* value:

i) if *RHL* = 0, discard the packet and omit the execution of further steps;

ii) if *RHL* > 0, update the field of the *Basic Header*, i.e. the *RHL* field with the decremented *RHL* value;

b) if no neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and SCF for the traffic class in the *TC* field of the *Common Header* is set, buffer the GAC packet in the *BC forwarding packet buffer* and omit the execution of further steps;

NOTE 2: If the GeoAdhoc router is outside the geographical area, the GN-PDU will not be passed to the upper layer entity.

11) if the return value of the forwarding algorithm is 0 (packet is buffered in a forwarding packet buffer) or -1 (packet is discarded), omit the execution of further steps;

12) execute media-dependent procedures; if the GN protocol constant `itsGnIfType` is set to:

a) UNSPECIFIED then no media-dependent procedures are specified;

b) for other values of `itsGnIfType` media dependent procedures may be defined elsewhere;

13) pass the GN-PDU to the ALI via the NET_ACC_DATA interface and set the destination address to the link layer address of the next hop LL_ADDR_NH.

10.4 MCO management

10.4.1 Overview

The Networking & Transport layer MCO functionalities are all part of the MCO_NET service. MCO_NET has interfaces with the GN Core, GN ALI group handlers, and the Facilities and Access layer functionalities.

The entities realizing MCO_NET required functionalities are (see figure 23):

- MCO Parameters Updates (MPU) (clause 10.4.2);
- Network Updates (NU) (clause 10.4.3);
- ALI Request Handler (ARH) (clause 10.4.4).

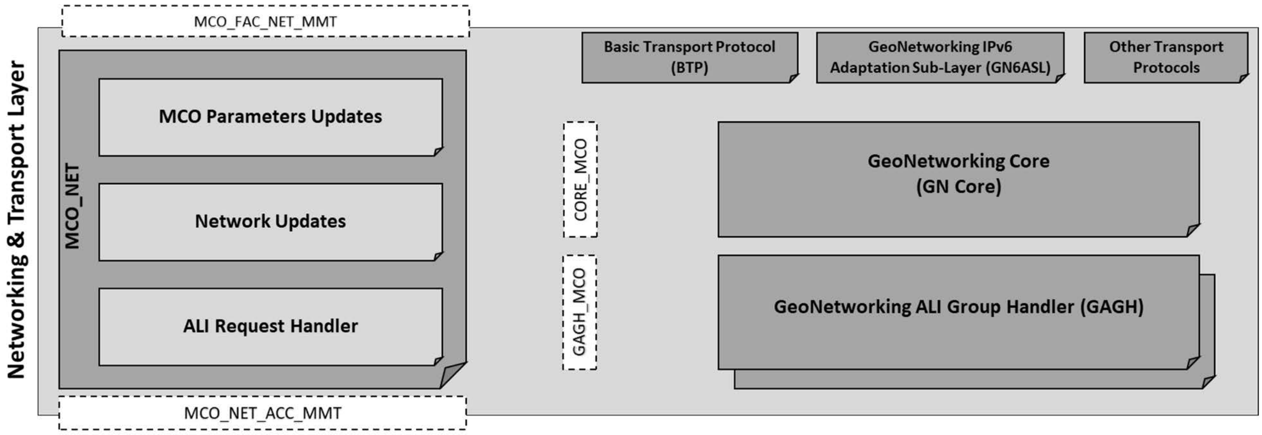

Figure 23: Overview of MCO management functions

10.4.2 MCO Parameters Updates (MPU)

10.4.2.1 Overview

Each ALI group at the Access layer is associated with a GAGH at the Networking & Transport layer. The state of a GAGH depends on the state of its associated ALI group (see ETSI TS 103 695 [i.7]). MCO_NET shall support MCO parameters update requests (e.g. to update an ALI group status) from the Facilities layer and shall provide to the Facilities layer MCO parameters updates indicated by the Access layer. The MPU entity handles MCO parameter requests from the Facilities layer and forwards them to the Access layer (figure 24). Additionally, the MPU entity handles MCO parameter updates from the Access layer. It forwards the MCO parameters updates to the Facilities layer and changes the states of the GAGHs.

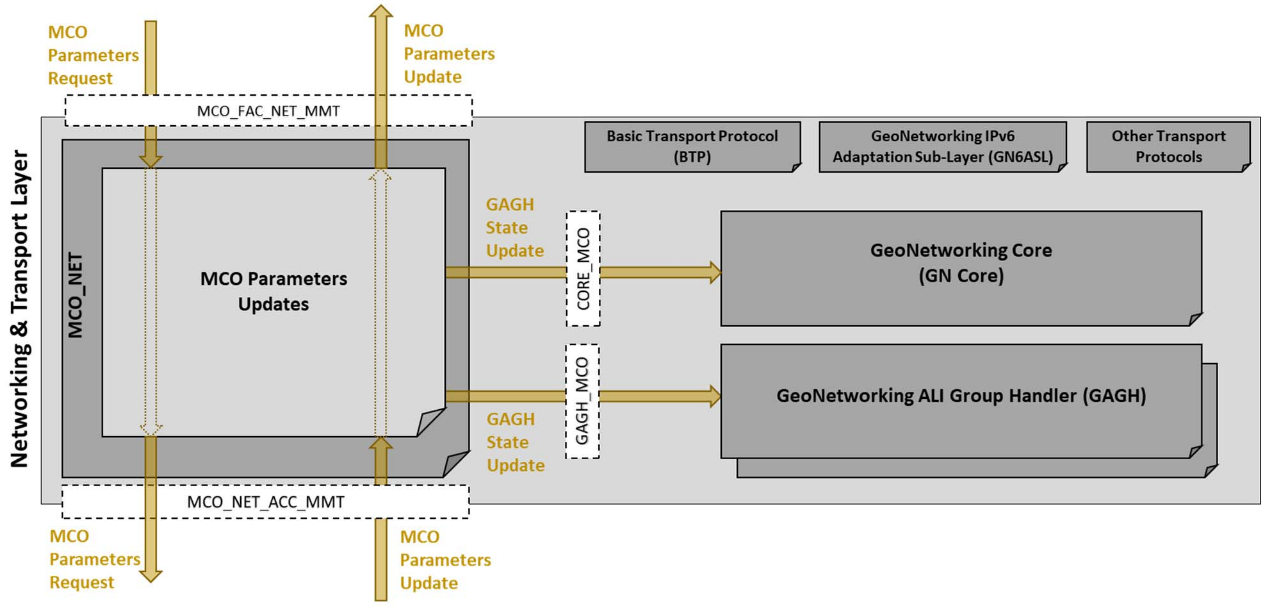

Figure 24: MCO parameters updates and flow control

10.4.2.2 GAGH state

A GAGH shall maintain a state machine with the following states:

- Active: an active GAGH performs GN protocol operations and maintains GN data structures and buffers. A GAGH is active as long its associated ALI group is in an active state.
- Inactive: an inactive GAGH does not perform GN protocol operations but maintains its data structures and buffers. A GAGH is inactive when its associated ALI group is in an inactive state.
- Disabled: a disabled GAGH does not perform GN protocol operations and has its data structures and buffers cleared. A GAGH is disabled when its associated ALI group is in an inactive or not available state.

NOTE: See ETSI TS 103 695 [i.7] for more information about the ALI group states.

10.4.2.3 GAGH state update

The MCO_NET shall provide the possibility to the Facilities layer to request MCO parameters updates at the lower layers and to receive MCO parameter updates from the lower layers.

When receiving a request to configure MCO parameters from the Facilities layer, the MPU shall forward the request to the Access layer. This can be realized via the MCO_NET_ACC_MMT interface (see ETSI TS 103 695 [i.7] for the specification of the interface).

When receiving an MCO parameter update from the Access layer, the MCO_NET shall forward the MCO parameters updates to the Facilities layer. If the state of an ALI group was modified, the MPU indicates the corresponding change on the GN_CORE entity via the CORE_MCO interface (annex L) and update the state of the associated GAGH via the GAGH_MCO interface (annex M).

10.4.3 Network Updates (NU)

10.4.3.1 Overview

To efficiently allocate resources at the Facilities layer, the MCO_NET shall provide network updates to the Facilities layer.

EXAMPLE: The network updates may contain the CLRs perceived locally and by neighboring stations. The information contained in the network update shall be media-independent.

The Access layer provides ALI group network updates. Information related to the GAGHs, such as the CLR, are passed to them by the NU entity. The NU entity may collect GAGH network updates containing perceived network conditions of neighboured stations. It then provides network updates to the Facilities layer (figure 25).

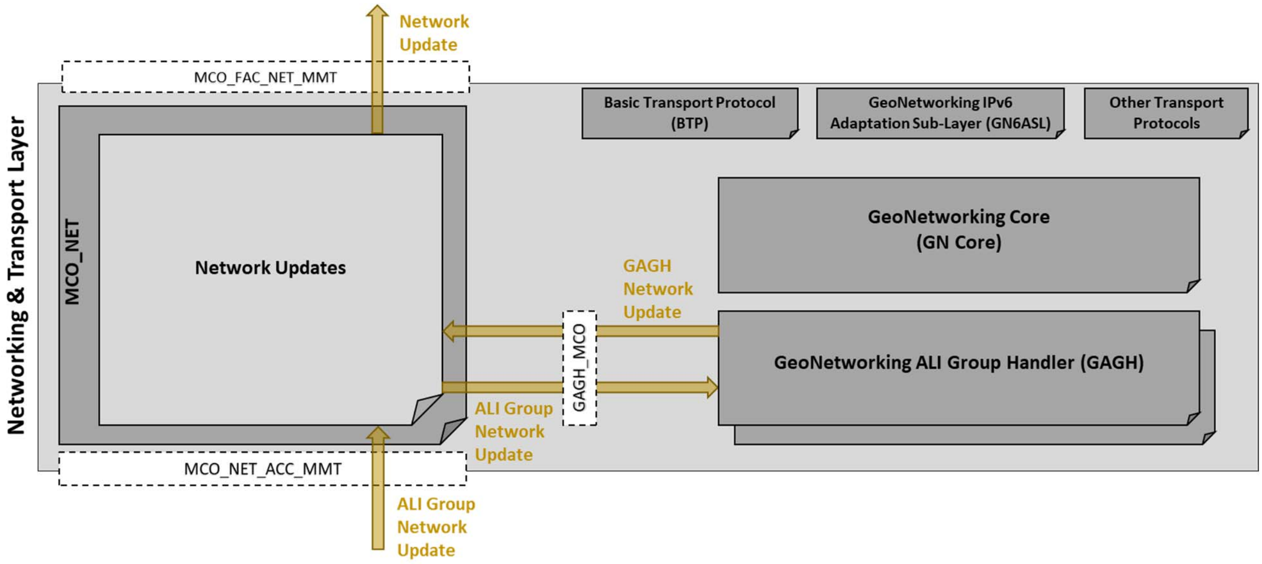

Figure 25: Network updates control flow

10.4.3.2 Access layer update

When receiving an Access layer update from the NET_ACC_MMT interface, the NU entity shall perform the following operations:

1) Send the CLR to the associated GAGH via the GAGH_MCO interface (annex M) when needed.

2) In case an Access Layer notification is indicated, the NU entity forwards the notification to the Facilities layer via the MCO_FAC_NET_MMT interface.

NOTE 1: More details of the MCO_FAC_NET_MMT interface are provided in annex N.

NOTE 2: The specifications of the media-dependent operations for GeoNetworking defines when the CLR should be sent.

10.4.3.3 Network updates to the Facilities layer

To provide better knowledge of the current communication capabilities, the NU entity collects network-related information from the GAGHs and the Access layer.

The network update provided by the NU entity to the Facilities layer via the MCO_FAC_NET_MMT interface shall be media-independent.

NOTE: To provide media-independent network updates to the Facilities layer, scaling functions may be used to align for example the CLR perceived by different access layer technologies, e.g. NR-V2X PC5 and ITS-G5.

10.4.4 ALI Request Handler (ARH)

When a packet dissemination is requested by the transport layer protocol entities, GN Core shall perform the following operations:

1) Check that the AliID indicated for the packet is associated to an active GAGH.

2) If the AliID is associated to an active GAGH, the GN Core sends the packet to the associated GAGH with the CORE_GAGH interface (see annex K).

3) If the AliID is not associated to an active GAGH (see figure 26) and:

   a) The upper protocol entity is BTP, the GN Core drops the packet and generates a *TRANS_CORE.confirm* with the result code *non-valid* AliID to the BTP entity (see annex J).

   b) The upper protocol entity is GN6ASL or any other transport protocols, the GN Core performs an ALI request to the ARH entity with the CORE_MCO interface (see annex L).

When the ARH entity receives an ALI request from the GN Core, the ARH entity shall perform the following operations:

1) The ARH transfers the ALI request to the Facilities layer entities via the MCO_FAC_NET_MMT interface (see annex N).

2) When receiving a response from the Facilities layer with the MCO_FAC_NET_MMT interface (see annex N) with an assigned AliID for an ALI request, the ARH transfers the response to the GN Core with the interface CORE_MCO (see annex L).

NOTE: For clarification, an AliID contains the information of which ALI group it belongs to. This can be used by the GN CORE entity to ensure that an active GAGH is associated to the ALI group referred in the AliID.

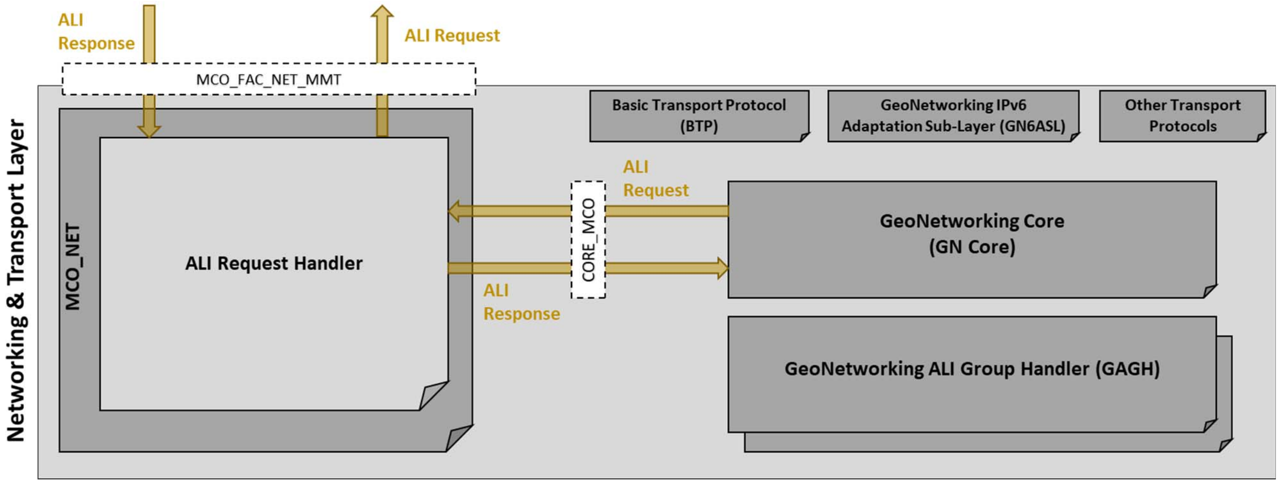

Figure 26: Handling of packet with no assigned ALI

Annex A (normative): Duplicate Packet Detection

A.1 General

A GAGH may receive multiple copies of the same packet. Reasons for packet duplications may be the forwarding of the packet from multiple GeoAdhoc routers, routing loops, misconfiguration or replay of packets from misbehaving GeoAdhoc routers. In order to control (e.g. prevent) the forwarding of duplicate packets, the GeoNetworking protocol uses mechanisms for Duplicate Packet Detection (DPD). The present document specifies a method for DPD that is based on sequence number (clause A.2).

The GeoNetworking protocol applies the sequence number-based method for DPD (clause A.2) to multi-hop packets (GUC, TSB, GBC, GAC, LS Request, LS Reply). This method is not applied to GeoNetworking single-hop packets (BEACON and SHB) since these packet types do not carry a SN field.

A.2 SN-based duplicate packet detection

For the SN-based method for DPD, a GAGH maintains a duplicate packet list DPL(GN_ADDR) for every entry in its LocT (see clause 8.2.2). When a GAGH processes a packet, the DPL is used to detect whether this packet is a duplicate. The DPL is a list with sequence numbers of a length `itsGnDPLLength`, which shall be much smaller than the maximum size of sequence numbers, i.e. 2^16-1. Optionally, the DPL may also contain a counter that indicates how often the packet with a particular sequence number has already been received from the source (duplicate packet counter, DPC).

When the GAGH processes a packet from source SO, it compares the value of the SN field carried in the GeoNetworking packet SN(P) and searches for the corresponding entry in the DPL. If SN(P) is already included in the DPL, the packet is marked as a duplicate. If the optional DPC is maintained, it is incremented. Otherwise, the new sequence number is added at the head of the DPL, overwriting the oldest entry.

NOTE: In one possible implementation, the DPL is a modified ring buffer (also called circular or cyclic buffer), i.e. a queue where the data to be added may wrap around to the beginning. The start of the ring buffer (head) contains the freshest newly received sequence number. The end of the queue (tail) contains the oldest received sequence number. Compared to a "classical" ring buffer, the DPL as a modified ring buffer is searched whether the received SN is already included.

The following algorithm shall be used:

```
-- DPL is the Duplicate Packet List for source SO with length DPL_Length
-- P is the received GeoNetworking packet
-- SN(P) is the sequence number in the received GeoNetworking packet
-- DPC is the duplicate packet counter for packet SN(P) from source SO, i.e. DPL(SN(P).DPC
If (SN(P) ∈ DPL)           # Received SN is already included in DPL for source SO
    Indicate packet as duplicate
    OPTIONAL: DPL(SN(P)).DPC++                  # Increment packet counter for received SN
ELSE                                            # Received SN is outside window
    Add SN(P)
ENDIF
```

Annex B (normative): Packet data rate and geographical area size control

B.1 Overview

Packet rate and geographical area size control is executed in the GeoNetworking forwarding process (clause 10.3) to control that a GAGH does not exceed a predefined packet rate or geographical area size.

B.2 Packet data rate control

A GAGH shall maintain the Exponential Moving Average (EMA) of the packet data rate PDR for every LocTE as calculated in equation (B.1).

Equation (B.1) Calculation of Exponential Moving Average of the packet data rate PDR:

$$PDR = \beta \times PDR_{t-1} + (1-\beta) \times x_t \qquad \text{(B.1)}$$

where:

x_t	is the measured instantaneous value of the packet data rate upon reception of the GeoNetworking packet.
PDR	is the average value of the packet data rate at time t; PDR_{t-1} is the previous value at time (t-1) maintained in the LocTE.
B	Weight factor (0 < ß < 1), set to the value of the GN protocol constant `itsGnMaxPacketDataRateEmaBeta / 100`.

If the packet data rate PDR of a GAGH exceeds the value of the GN protocol constant `itsGnMaxPacketDataRate,` packets from this GeoAdhoc router (source or sender) shall not be forwarded.

B.3 Geographical area size control

If the geographical area size carried in a GBC or GAC packet exceeds the maximum value specified in the GN protocol constant `itsGnMaxGeoAreaSize`, the GeoNetworking packet shall not be sent by the source and shall not be forwarded by the forwarder.

Annex C (normative): Position vector update

C.1 Overview

The position vector update is executed in the GeoNetworking forwarding process (clause 10.3) when a PV in a LocTE is updated by PV carried in a GeoNetworking packet header. The algorithm ensures that always the newer PV is used indicated by the timestamp that is contained in the PV.

The algorithm is utilized in two cases:

1) When a GeoNetworking packet is received, the forwarding procedure updates the PV in the LocT by the PV carried in the GeoNetworking packet.

2) When a GeoNetworking packet is forwarded, the forwarding procedure updates the PV in the packet to be forwarded by the PV in the LocT.

The algorithm makes use of the timestamp that is associated with the position information and is part of the position vector fields. It handles the wraparound of the timestamp values that occur due to the limited number of bits that represent a timestamp.

NOTE: With a 32 bit timestamp in [ms], a wraparound occurs after 50 days: $(((((2^{32}) - 1) / 1\,000) / 60) / 60) / 24 = 49{,}7102696$.

C.2 Update of LocT position vector

The following algorithm shall be applied to update a PV in the LocT. The algorithm shall also reset the lifetime of the *location table entry T(LocTE)* (clause 8.2.3).

```
-- RP is the received GeoNetworking packet
-- PV_RP is the position vector in the received GeoNetworking packet
-- PV_LocT is the position vector in the LocT to be updated
-- TST_PV,RP is the timestamp for the position vector in the received
--      GeoNetworking packet
-- TST_PV,LocT is the timestamp for the position vector in the location table
--      to be updated
-- TS_Max is the maximum value of the timestamp = 2^32-1
-- T(LocTE)is the lifetime of the location table entry
-- itsGnLifetimeLocTE is the value of the GN protocol constant itsGnLifetimeLocTE
IF (((TST_PV,RP > TST_PV,LocT) AND ((TST_PV,RP – TST_PV,LocT) <= TST_Max/2)) OR
     ((TST_PV,LocT > TST_PV,RP) AND ((TST_PV,LocT – TST_PV,RP) > TST_Max/2))) THEN
    TST_PV,RP is greater than TST_PV,LocT
    PV_LocT ← PV_RP
    T(LocTE) ← value(itsGnLifetimeLocTE)
ELSE
        TST_PV,RP is not greater than TST_PV,LocT
ENDIF
```

C.3 Update of GeoNetworking packet position vector

The following algorithm shall be applied to update a PV in a packet to be forwarded:

```
-- FP is the GeoNetworking packet to be forwarded
-- PV_{FP} is the position vector in the GeoNetworking packet to be forwarded
-- PV_{LocT} is the position vector in the LocT
-- TST_{PV,FP} is the timestamp for the position vector in the GeoNetworking
--        packet to be forwarded
-- TST_{PV,LocT} is the timestamp for the position vector in the location table
-- TS_{Max} is the maximum value of the timestamp = 2^32-1
IF (((TST_{PV,LocT} > TST_{PV,FP}) AND ((TST_{PV,LocT} – TST_{PV,FP}) <= TST_{Max}/2)) OR
((TST_{PV,FP} > TST_{PV,LocT}) AND ((TST_{PV,FP} – TST_{PV,LocT}) > TST_{Max}/2))) THEN
        # TST_{PV,LocT} is greater than TST_{PV,FP}
        PV_{FP} ← PV_{LocT}
ELSE
        # TST_{PV,FP} is not greater than TST_{PV,LocT}
ENDIF
```

NOTE: The algorithm is used in to update the DE PV fields with the PV(DE) in the LocT (clause 10.3.8.3 Forwarder operations for GUC packet handling). It determines whether the timestamp of the PV in the LocT is fresher than the one in the GeoNetworking packet taking into account wrapping of the timestamp.

Annex D (normative): GeoNetworking forwarding algorithm selection procedure

The procedure utilizes the function F(x,y) specified in ETSI TS 103 899 [1] clause 5 in order to determine whether the GeoAdhoc router is located inside, at the border or outside of the geographical target area carried in the GeoBroadcast or GeoAnycast packet header.

If the GeoAdhoc router is inside or at the border of the area, the packet shall be processed according to the selected *area forwarding* algorithm (`itsGnAreaForwardingAlgorithm`).

If the GeoAdhoc router is outside of the area and the packet is received from a sender located outside of the area, the packet shall be forwarded according to the selected *non-area forwarding* algorithm (`itsGnNonAreaForwardingAlgorithm`).

If the GeoAdhoc router is outside of the area and the packet is received from a sender located inside or at the border of the area, the packet shall be discarded.

NOTE 1: Packet duplicate detection is not part of the forwarding algorithm but of the packet handling operations.

NOTE 2: As defined in ETSI TS 103 899 [1] clause 5:

$$F(x,y)=\begin{cases}=1 & \text{for } x=0 \text{ and } y=0 \text{ (at the centre point)}\\ >0 & \text{inside the geographical area}\\ =0 & \text{at the border of the geographical area}\\ <0 & \text{outside the geographical area}\end{cases} \qquad \text{(D.1)}$$

The algorithm returns one of the following three values:

- AREA-FORWARDING indicates that forwarding continues with one of the algorithms defined in annex F selected by `itsGnAreaForwardingAlgorithm`;
- NON-AREA-FORWARDING indicates that forwarding continues with one of the algorithms defined in annex E selected by `itsGnNonAreaForwardingAlgorithm`;
- -1 indicates that the packet is discarded.

The pseudo-code of the algorithm is below:

```
-- P is the GeoNetworking packet to be forwarded
-- LAT_EGO and LONG_EGO are latitude and longitude of the EPV, respectively
-- PV_SE is the sender position vector in the LocTE for the sender of P
--      with LAT_SE and LONG_SE as latitude and longitude
--      and position accuracy indicator PAI_SE
-- SE_POS_VALID is true if sender position is valid (known and accurate)

SET F_EGO ← F(LAT_EGO, LONG_EGO)    # Eq. D.1 in NOTE 2

IF (F_EGO ≥ 0) THEN
    # Local GeoAdhoc router is inside or at the border of target area
    RETURN AREA-FORWARDING          # Continue with area forwarding algorithm in annex F
ELSE
    # Local GeoAdhoc router is outside of target area
    SET F_SE ← F(LAT_SE, LONG_SE)   # Eq. D.1 in NOTE 2
    SET SE_POS_VALID ← PV_SE EXISTS AND PAI_SE = TRUE
    IF (SE_POS_VALID AND F_SE ≥ 0)THEN
        # Prevents transition from area forwarding to non-area forwarding
        RETURN -1                   # P is to be discarded
    ELSE
        RETURN NON-AREA-FORWARDING  # Continue with non-area forwarding algorithm in annex E
    ENDIF
ENDIF
```

The activity diagram of the forwarding algorithm selection procedure is depicted in figure D.1 for illustration.

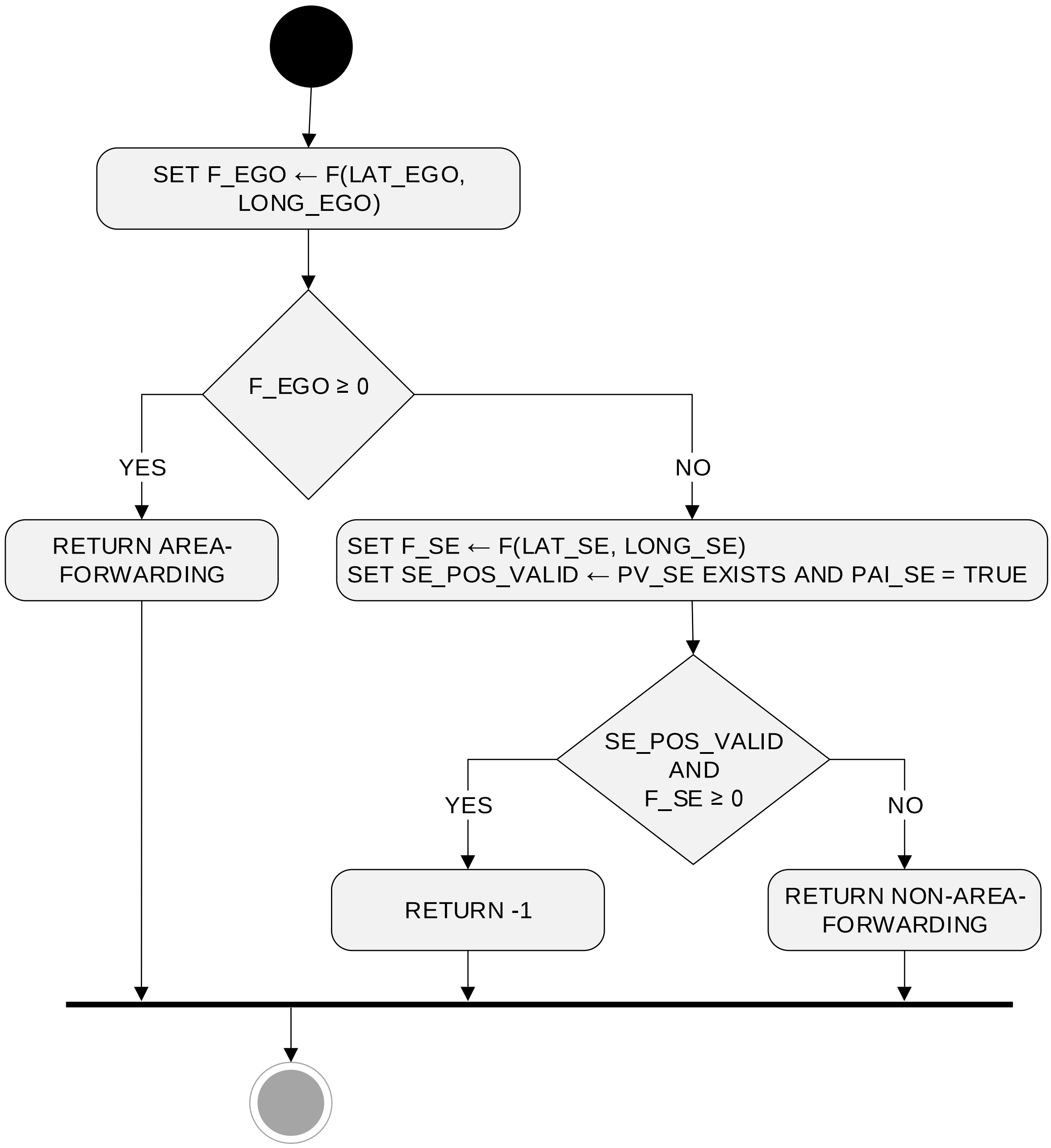

Figure D.1: GeoNetworking forwarding algorithm selection procedure

Annex E (normative): Non-area forwarding algorithms

E.1 Overview

The non-area forwarding algorithms are used to route a packet towards a destination. It is executed by a GeoAdhoc router to relay a packet to the next hop.

The present document defines two non-area forwarding algorithms:

1) Greedy Forwarding (GF) algorithm (clause E.2);

2) Contention-Based Forwarding (CBF) algorithm (clause E.3).

E.2 Greedy forwarding algorithm

With the Greedy Forwarding (GF) algorithm, the GeoAdhoc router uses the location information of the destination carried in the GN packet header and selects one of the neighbours as the next hop.

The algorithm applies the *most forward within radius (MFR)* policy, which selects the neighbour with the smallest geographical distance to the destination, thus providing the greatest progress when the GN packet is forwarded.

The algorithm returns one of the following two values:

- the link layer address of the next hop LL_ADDR_NH;
- 0 indicates that no forwarder could be found and the packet is buffered in the appropriate *forwarding packet buffer*.

The pseudo-code of the algorithm is below:

```
-- P is the GN packet to be forwarded
-- I is the i-th LocTE
-- NH is the LocTE identified as next hop, NH.ALI_ADDR its Access layer instance address
-- NH_ALI_ADDR is the Accesss layer instance address of the next hop
-- EPV is the ego position vector
-- PV_P is the destination position vector in the GeoNetworking packet to be forwarded
-- PV_I is the position vector of the i-th LocTE
-- MFR indicates the progress according to the MFR policy
-- B is the forwarding packet buffer
   (UC forwarding buffer or BC forwarding buffer, depending on type of P)
-- TC is the traffic class of the GN-Data.request (source operations)
   or the field in the received Common header (forwarder operations)
MFR = DIST(PV_P, EPV)                              Initialize MFR
FOR (i∈ LocT)
    IF (i.IS_NEIGHBOUR) THEN                       # LocTE i is neighbour
        IF (DIST(PV_P, PV_I) < MFR) THEN
            NH ← i
            MFR ← DIST(PV_P, PV_I)
        ENDIF
    ENDIF
ENDFOR
IF (MFR < DIST(PV_P, EPV)) THEN
    SET NH_ALI_ADDR ← NH.ALI_ADDR
ELSE                                               # Forwarder is at a local optimum
    IF (TC.SCF_IS_ENABLED) THEN
        ADD P TO B
        SET NH_ALI_ADDR ← 0                        # Indicates that packet is buffered
    ELSE
        SET NN_ALI_ADDR ← BCAST                    # No buffering allowed, fall back to BCAST
    ENDIF
ENDIF
RETURN NH_ALI_ADDR
```

NOTE: If no neighbour with greater progress than the local GeoAdhoc router exists, i.e. no suitable neighbour exists, the packet has reached a local optimum and the result is returned indicating that no forwarder could be found.

E.3 Non-area contention-based forwarding algorithm

With the Contention-Based Forwarding (CBF) algorithm, a receiver decides to be a forwarder of a GN packet. This is contrary to the sender-based forwarding scheme specified in clause E.2, where the sender determines the next hop. The CBF algorithm utilizes timer-based re-broadcasting with overhearing of duplicates in order to enable an implicit forwarding of a packet by the optimal node.

With CBF, the GeoAdhoc router broadcasts the GN packet. All neighbours, which receive the packet, process it: those routers with a positive progress buffer the packet in the *CBF packet buffer* and start a timer with a timeout that is inversely proportional to the forwarding progress of the GeoAdhoc router (equation (E.1)).

Equation (E.1) Calculation of timeout TO_CBF for buffering packets in the *CBF packet buffer*:

$$TO\_CBF = \begin{cases} TO\_CBF\_MAX + \dfrac{TO\_CBF\_MIN - TO\_CBF\_MAX}{DIST\_MAX} \times PROG & \text{for } PROG \le DIST\_MAX \\ TO\_CBF\_MIN & \text{for } PROG > DIST\_MAX \end{cases} \quad \text{(E.1)}$$

where:

TO_CBF_MIN is the minimum duration the packet shall be buffered in the *CBF packet buffer*.

TO_CBF_MAX is the maximum duration the packet shall be buffered in the *CBF packet buffer*.

PROG is the forwarding progress of the local GeoAdhoc router towards the destination, i.e. the difference between the sender's distance and GeoAdhoc router's local distance from the destination. The sender position is taken from its LocTE.

DIST_MAX is the theoretical maximum communication range of the wireless access technology.

NOTE 1: For PROG = DIST_MAX, TO_CBF becomes TO_CBF_MIN. For the (theoretical) PROG = 0, TO_CBF becomes TO_CBF_MAX.

TO_CBF_MIN and TO_CBF_MAX shall be set to the GN protocol constants `itsGnCbfMinTime` and `itsGnCbfMaxTime`, respectively. If DIST_MAX is not defined in the specification of GeoNetworking media-dependent functionality for the specific ITS access technology, it shall be set to the GN protocol constant `itsGnDefaultMaxCommunicationRange`.

Upon expiration of the timer, the GeoAdhoc router re-broadcasts the GN packet. Before the timer expires, the GeoAdhoc router may receive a duplicate of the packet from a GeoAdhoc router with a shorter timeout, i.e. with a smaller distance to the destination. In this case, the GeoAdhoc router inspects its *CBF packet buffer*, stops the timer and removes the GN packet from the *CBF packet buffer*.

NOTE 2: Compared to the GF algorithm (clause E.2), CBF has an implicit reliability mechanism at the cost of larger forwarding delay and additional processing. The reliability mechanism ensures that a packets is re-forwarded by an alternative forwarder if the theoretically optimal forwarder does not receive the packet, e.g. due to wireless link errors.

The algorithm returns one of the following three values:

- the Broadcast ALI address BCAST;
- 0 indicates that the packet is buffered in the *CBF packet buffer* and will further processed when the timer expires or a packet duplicate is handled;
- -1 indicates that the packet is discarded.

The activity diagram of the CBF algorithm is depicted in figure E.1 for illustration.

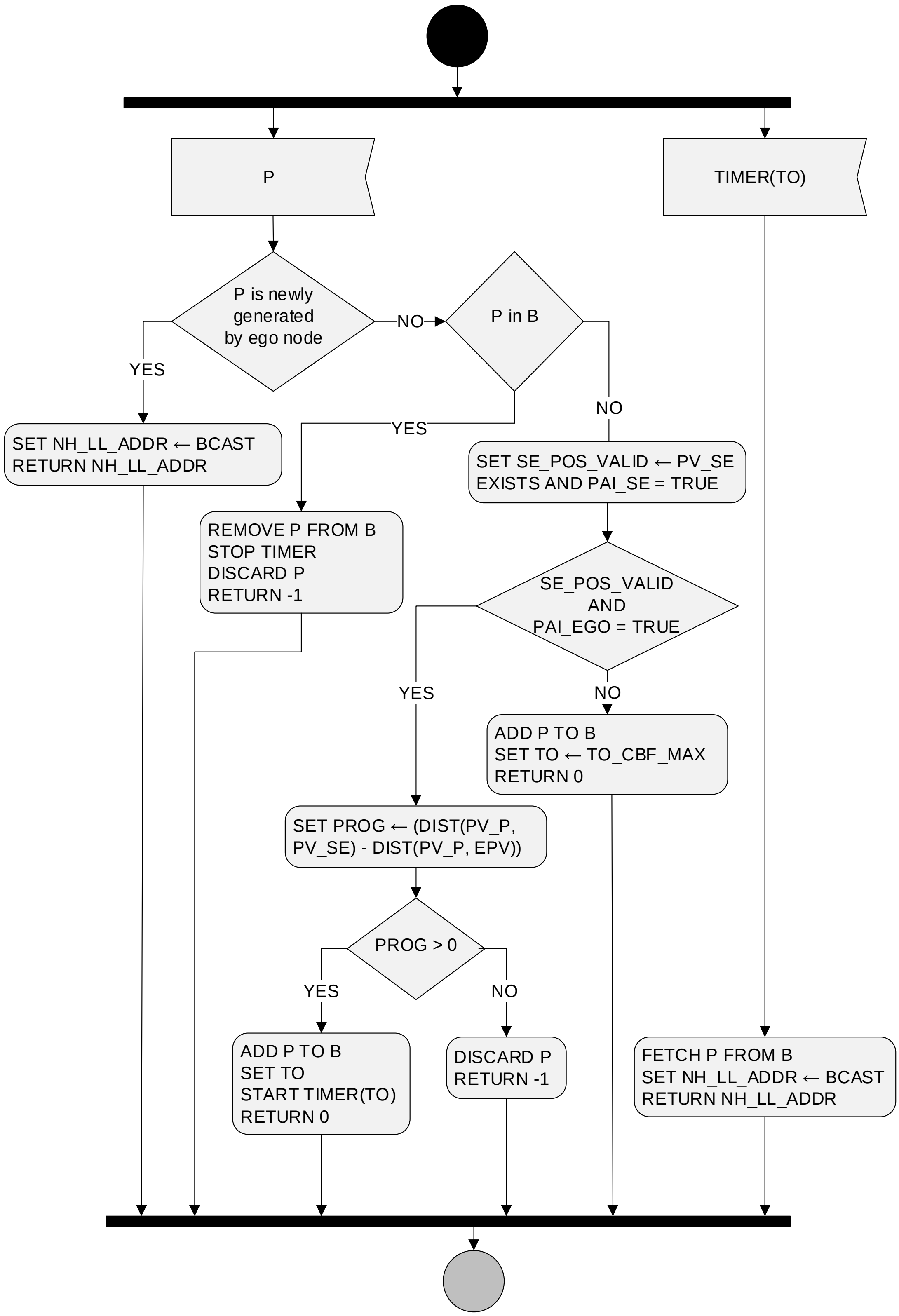

Figure E.1: Non-area contention-based forwarding algorithm

The pseudo-code of the algorithm is below:

```
-- P is the GN packet to be forwarded
-- EPV is the ego position vector with position accuracy indicator PAI_EGO
-- PV_P is DE PV of GUC header or destination area's centre position for GBC/GAC header
-- PV_SE is the sender position vector in the LocT with position accuracy indicator PAI_SE
-- B is the CBF packet buffer
-- TO is the timeout that triggers the re-broadcast of the packet
-- NH_ALI_ADDR is the ALIaddress of the next hop
-- BCAST is the Broadcast LL address

IF (P is newly generated by ego node) THEN
    SET NH_LL_ADDR ← BCAST
    RETURN NH_LL_ADDR                           # Immediate transmission at source router

ELSE IF (P IN B) THEN                           # Contending
        REMOVE P FROM B
        STOP TIMER
        DISCARD P
        RETURN -1                               # Indicates that packet is discarded
ELSE                                            # New packet
    SET SE_POS_VALID ← PV_SE EXISTS AND PAI_SE = TRUE
                                                # Indicates that sender position is valid
    IF (SE_POS_VALID AND PAI_EGO = TRUE) THEN
        # Sender and ego position are reliable, can calculate PROG
        SET PROG ← (DIST(PV_P, PV_SE)-- DIST(PV_P, EPV))
        IF (PROG > 0) THEN                      # Forwarding progress
            ADD P TO B
            SET TO ← TO_CBF(PROG)               # Eq. E.1 in the present clause
            START TIMER(TO)
            RETURN 0                            # Indicates that packet is buffered
        ELSE
            DISCARD P
            RETURN -1                           # Indicates that packet is discarded
        ENDIF
    ELSE
        # Use maximum timer if PROG cannot be calculated reliably
        ADD P TO B
        SET TO ← TO_CBF_MAX
        START TIMER(TO)
        RETURN 0                                # Indicates that packet is buffered
    ENDIF
ENDIF

IF (TIMER(TO) EXPIRES) THEN
    FETCH P FROM B                              # Removes P from B
    SET NH_LL_ADDR ← BCAST
    RETURN NH_LL_ADDR                           # Indicates that packet can be forwarded
ENDIF
```

Annex F (normative): Area forwarding algorithms

F.1 Overview

The GeoBroadcast forwarding algorithm is used to distribute a data packet within a geographical target area. The present document defines three forwarding algorithms:

1) Simple area forwarding algorithm (clause F.2).

2) Contention-based area forwarding algorithm (clause F.3).

3) Advanced GeoBroadcast forwarding algorithm (clause F.4).

The area forwarding algorithms assume that the sender of the data packet is located inside or at the border of the target area. If this is not the case, the packet can be transported from the sender towards the target area, i.e. using non-area forwarding algorithms as specified in annex E, which is also referred to as line forwarding.

F.2 Simple GeoBroadcast forwarding algorithm

The algorithm applies when the GeoAdhoc router is located inside or at the border of the geographical target area carried in the GeoBroadcast packet header. In this case, the packet shall be re-broadcasted.

NOTE 1: Packet duplicate detection is not part of the forwarding algorithm but of the packet handling operations.

NOTE 2: The algorithm described in annex D determines whether the present algorithm applies.

The algorithm returns the Broadcast LL address BCAST.

F.3 Area contention-based forwarding algorithm

Similar to the non-area contention-based forwarding algorithm, with the area contention-based forwarding (CBF) algorithm, a receiver decides to be a forwarder of a GN packet.

The algorithm applies when the GeoAdhoc router is located inside or at the border of the geographical target area carried in the GeoBroadcast or GeoAnycast packet header. When a node broadcasts a GBC/GAC packet with the CBF algorithm, all receiving neighbours process and buffer this packet in their respective *CBF packet buffer*. The associated CBF timer starts with a timeout that is proportional to the distance between the GeoAdhoc router's local position and the position of the sender. Thus, the node with the maximum forwarding progress will have the smallest timeout (equation (F.1)). When the timer expires the node will re-broadcast the GBC packet and implicitly inform the GeoAdhoc routers in its communication range to not forward the packet. Upon reception of the duplicate these GeoAdhoc routers stop the timer and remove the packet from the *CBF packet buffer*.

NOTE 1: The definition of the distance is different from the non-area CBF algorithm (clause E.3), where the distance is the forwarding progress between the GeoAdhoc router's local position and the destination position.

NOTE 2: Procedure described in annex D determines whether the present algorithm applies.

Equation (F.1): Calculation of timeout TO_CBF for buffering packets in the *CBF packet buffer*:

$$TO\_CBF = \begin{cases} TO\_CBF\_MAX + \frac{TO\_CBF\_MIN -- TO\_CBF\_MAX}{DIST\_MAX} \times DIST & for\ DIST \le DIST\_MAX \\ TO\_CBF\_MIN & for\ DIST > DIST\_MAX \end{cases} \quad (F.1)$$

where:

TO_CBF_MIN is the minimum duration the packet shall be buffered in the *CBF packet buffer*.

TO_CBF_MAX is the maximum duration the packet shall be buffered in the *CBF packet buffer*.

DIST is the distance between the GeoAdhoc router's local position and the sender (i.e. previous forwarder or source) position. The sender position is taken from its LocTE.

DIST_MAX is the theoretical maximum communication range of the wireless access technology.

NOTE 3: For DIST = DIST_MAX, TO_CBF_GBC becomes TO_CBF_MIN. For the (theoretical) distance DIST = 0, TO_CBF_GBC becomes TO_CBF_MAX.

TO_CBF_MIN and TO_CBF_MAX shall be set to the GN protocol constants `itsGnCbfMinTime` and `itsGnCbfMaxTime`, respectively. If DIST_MAX is not defined in the specification of GeoNetworking media-dependent functionality for the specific ITS access technology, it shall be set to the GN protocol constant `itsGnDefaultMaxCommunicationRange`.

Upon expiration of the timer, the GAGH of the GeoAdhoc router re-broadcasts the GBC/GAC packet. Before the timer expires, the GAGH may receive a duplicate of the packet from a GeoAdhoc router with a shorter timeout, i.e. with a smaller distance to the destination. In this case, the GAGH inspects its *CBF packet buffer*, stops the timer and removes the GBC/GAC packet from the *CBF packet buffer*.

The activity diagram of the area CBF algorithm is depicted in figure F.1 for illustration.

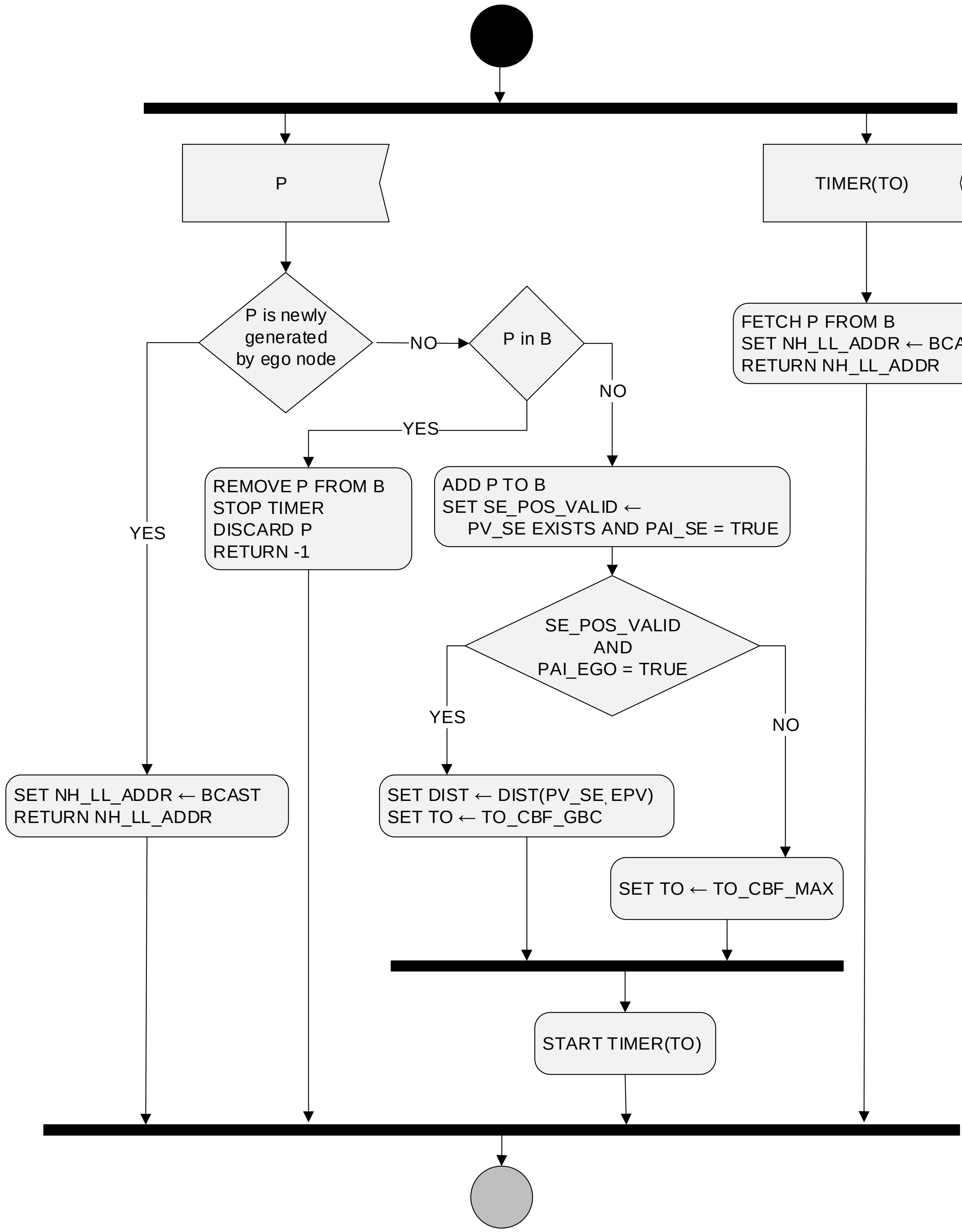

Figure F.1: Area contention-based forwarding algorithm

The algorithm returns one of the following four values:

- the Broadcast LL address BCAST;
- the LL address of the next hop NH_LL_ADDR;
- 0 indicates that the packet is buffered in the *CBF buffer*;

- -1 indicates that the packet is discarded.

The pseudo-code of the algorithm is below:

```
-- P is the GN packet to be forwarded
-- EPV is the ego position vector with position accuracy indicator PAI_EGO
-- PV_SE is the sender position vector in the LocTE with position accuracy indicator PAI_SE
-- B is the CBF packet buffer
-- TO is the timeout that triggers the re-broadcast of the packet
-- NH_LL_ADDR is the LL address of the next hop
-- BCAST is the Broadcast LL address

IF (P is newly generated by ego node) THEN
    SET NH_LL_ADDR ← BCAST
    RETURN NH_LL_ADDR                   # Immediate transmission by source router
ELSE IF (P IN B) THEN                   # Contending
    REMOVE P FROM B
    STOP TIMER
    DISCARD P
    RETURN -1                           # Indicates that packet is discarded
ELSE                                    # New packet
    ADD P TO B
    SET SE_POS_VALID ← PV_SE EXISTS AND PAI_SE = TRUE
                                        # Indicates that sender position is valid
    IF (SE_POS_VALID AND PAI_EGO = TRUE) THEN
        SET DIST ← DIST(PV_SE, EPV)
        SET TO ← TO_CBF(DIST)           # Eq. F.1 in the present clause
    ELSE
        # Use maximum timer if DIST cannot be calculated reliably
        SET TO ← TO_CBF_MAX
    ENDIF
    START TIMER(TO)
    RETURN 0                            # Indicates that packet is buffered
ENDIF

IF (TIMER(TO) EXPIRES) THEN
    FETCH P FROM B
    SET NH_LL_ADDR ← BCAST
    RETURN NH_LL_ADDR
ENDIF
```

F.4 Area advanced forwarding algorithm

The area advanced forwarding algorithm includes mechanisms from the Greedy Forwarding (GF) algorithm (clause E.2) and the area Contention-Based Forwarding (CBF) algorithm (clause F.3). As such it is both sender-based and receiver-based. It also includes further enhancements of CBF in order to improve the efficiency and reliability.

The relies on four main mechanisms:

1) CBF is used to deal with uncertainties in terms of reception failure caused by mobility of ITS-S, fading phenomena and collisions on the wireless medium.

2) In order to minimize the additional forwarding delay introduced by CBF, CBF is complemented with the selection of one specific forwarder, referred to as next hop, at the sender. Upon reception of the packet, the next hop - in case of correct reception - forwards the message immediately.

3) The efficiency of CBF is improved by choosing potential forwarders only from a specific sector of the circular forwarding area; i.e. GeoAdhoc routers located inside the sector (defined by an angle and the maximum communication range) refrain from retransmission of the packet (sectorial backfire).

4) The reliability of the dissemination process is increased by a controlled packet retransmission scheme within the geographical target area.

The algorithm returns one of the following four values:

- the LL address of the next hop (NH_LL_ADDR);
- the Broadcast LL address (BCAST);
- 0 indicates that the packet is buffered in the *CBF buffer*;
- -1 indicates that the packet is discarded.

The algorithm applies when the GeoAdhoc router is located inside or at the border of the geographical target area carried in the GeoBroadcast packet header. At the source, the algorithm selects the next forwarder from its location table, forwards the packet to the neighbour with the greatest progress (GF) and additionally enters CBF mode (i.e. buffers the packet in the *CBF buffer* and starts a timer). When a GAGH of a GeoAdhoc router receives a packet, it checks whether it is located inside/at the border of the area. If so and the packet is received as a unicast link layer frame (i.e. the GeoAdhoc router was selected as next hop by the sender of the packet), it again forwards the packet by GF. Otherwise, the GAGH checks whether it is already contending (i.e. the packet is already in the *CBF buffer*). In this case, the packet is regarded as a duplicate and the GAGH of the GeoAdhoc router counts how often the packet is received and where the sender of the duplicate is located: If the counter exceeds a threshold (COUNTER ≥ MAX_COUNTER) and the local GeoAdhoc router is inside/at the border of the sectorial area, the contention is stopped and the packet is discarded.

For the sectorial backfire (mechanism 3, see above), the algorithm uses a function G with the properties in equation (F.2) and specified in equation (F.3) in order to determine whether the GeoAdhoc router is located inside, at the border or outside a sectorial area that is defined by the sender's position, the distance between sender and local GeoAdhoc router, the distance between the sender and the forwarder, the (theoretical) maximum communication range of the wireless technology and an angle between the forwarder, the sender and the local GeoAdhoc router positions (figure F.2). In principle, if a GeoAdhoc router is contenting in CBF mode and located outside the sectorial area, the packet is scheduled for re-broadcast. If a GeoAdhoc router is located inside the sectorial area, it refrains from contending.

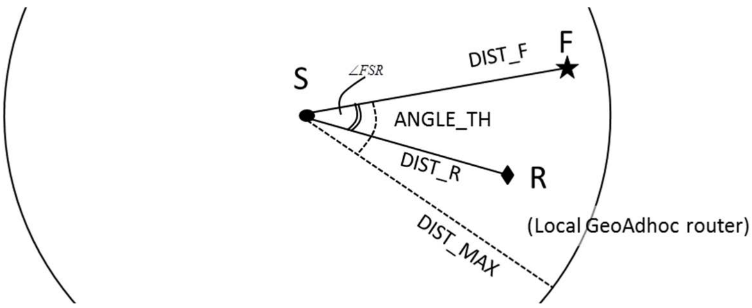

Figure F.2: Sectorial contention area

Equation (F.2): Properties of the geometric function G:

$$G = \begin{cases} +1 & \text{inside or at the border of the sectorial area} \\ -1 & \text{outside the sectorial area} \end{cases} \tag{F.2}$$

Equation (F.3): Calculation of the sectorial contention area:

$$G = \begin{cases} +1 & \text{for } (DIST\_R < DIST\_F) \text{ AND } (DIST\_F < DIST\_MAX) \text{ AND } (\angle FSR \leq ANGLE\_TH) \\ -1 & \text{Otherwise} \end{cases} \tag{F.3}$$

where:

DIST_R is the distance between the GeoAdhoc router's local position and the sender position. The sender position is taken from GeoAdhoc router's local LocTE.

DIST_F is the distance between the forwarder position and the sender's position. The forwarder and sender positions are taken from the corresponding LocTE of the local GeoAdhoc router.

DIST_MAX is the theoretical maximum communication range of the wireless access technology.

$\angle FSR$ is the angle between the positions of the forwarder, the sender and the local GeoAdhoc router.

ANGLE_TH is a threshold value for the angle. This threshold shall have a minimum and a maximum value of 30° and 60°, respectively. It shall vary accordingly to neighbour node density and the default value is given by the GN protocol constant itsGnBroadcastCBFDefSectorAngle.

NOTE: In a possible implementation, the neighbour nodes density depends on the neighbour density, which is defined by the number of neighbour nodes seen by the local GeoAdhoc router over the geographical area covered by the theoretical maximum communication range of the wireless access technology, see the examples 1, 2 and 3 below.

EXAMPLE 1: When DEN_NEIGH < 0,025 node/m^2 → ANGLE_TH = 30°

EXAMPLE 2: When 0,025 node/m^2 < DEN_NEIGH < 0,05 node/m^2 → ANGLE_TH = 45°

EXAMPLE 3: When DEN_NEIGH ≥ 0,05 node/m^2 → ANGLE_TH = 60°

In order to increase the reliability of the dissemination process by controlled packet retransmission, a GeoAdhoc router in CBF mode maintains a counter for the number of re-transmissions for a packet. This counter is incremented every time this packet is received. When the number of re-transmissions for this packet reaches a threshold, the GeoAdhoc router stops contending for the packet. By this mechanism, the packet is allowed to be re-transmitted several times for better reliability, but the data overhead is controlled.

The activity diagram of the area advanced forwarding algorithm is illustrated in figure F.3.

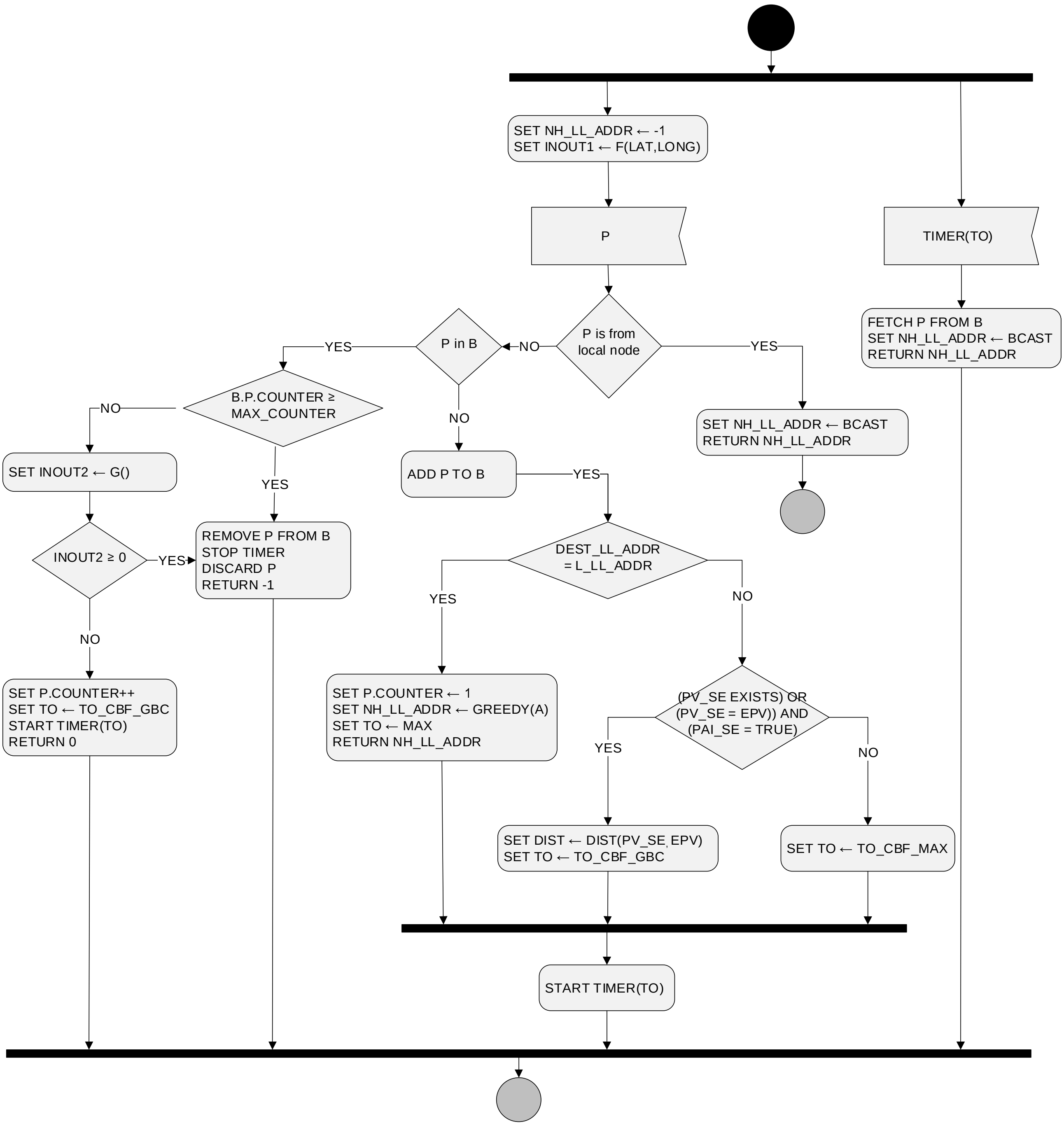

Figure F.3: Area advanced forwarding activity diagram

The pseudo-code of the algorithm is below:

```
-- P is the GBC packet to be forwarded
-- L_LL_ADDR is the LL address of the GAGH of the local GeoAdhoc router
-- NH_LL_ADDR is the LL address of the next hop
-- DEST_LL_ADDR is the LL destination address carried in P
-- B is the CBF packet buffer
-- EPV is the local position vector with latitude LAT and longitude LONG
-- PV_SE is the sender position vector in its LocTE with latitude LAT_SE, longitude LONG_SE
--    and position accuracy indicator PAI_SE
-- TO is the timeout that triggers the re-broadcast of the packet
-- COUNTER is the retransmit counter for the packet P
-- MAX_COUNTER is the retransmit threshold
-- BCAST is the Broadcast LL address
-- GREEDY() is the GF algorithm as specified in clause E.2
-- INOUT1 indicates whether the local GeoAdhoc router is outside the target area or not
-- INOUT2 indicates whether the local GeoAdhoc router is outside the sectorial contention
--    area or not.
-- INOUT3 indicates whether the sender is outside the target area or not

IF (P is from local node) THEN          # If local node is origin, can send directly
    SET NH_LL_ADDR ← BCAST
    RETURN NH_LL_ADDR
END
SET NH_LL_ADDR ← -1                     # Initialize NH_LL_ADDR
SET INOUT1 ← F(LAT,LONG)                    # Eq. D.1 in Note 2 of annex D
IF (P IN B) THEN                            # Contending
    IF (B.P.COUNTER ≥ MAX_COUNTER) THEN # Stop contending
        REMOVE P FROM B                     # Remove packet from CBF buffer
        STOP TIMER
        DISCARD P                           # Discard packet
        RETURN -1                           # Indicates that packet is discarded
    ELSE
        SET INOUT2 ← G()                    # Eq. E.4 for sectorial contention area
        IF (INOUT2 ≥ 0) THEN                # Inside or at the border of sectorial area
            REMOVE P FROM B                 # Remove packet from CBF buffer
            STOP TIMER
            DISCARD P                       # Discard packet
            RETURN -1                       # Indicates that packet is discarded
        ELSE                                # Outside of sectorial area
            SET COUNTER++
            SET TO ← TO_CBF_GBC         # Eq. F.1 in clause F.3
            START TIMER(TO)
            RETURN 0                        # Indicates that packet is buffered
        ENDIF
    ENDIF
ELSE                                        # New packet
    ADD P TO B
    IF (DEST_LL_ADDR = L_LL_ADDR) THEN      # Greedy forwarding
        SET P.COUNTER ← 1                   # Initialize COUNTER
        SET NH_LL_ADDR ← GREEDY(A)      # Greedy()returns LL address of next hop or 0
        SET P.TO ← MAX                      # Set to TO_CBF_MAX (F.1 in clause F.3)
        START TIMER(TO)
        RETURN NH_LL_ADDR
    ELSE                                    # CBF
        IF ((PV_SE EXISTS) OR (PV_SE = EPV)) AND (PAI_SE = TRUE)) THEN
            SET DIST ← DIST(PV_SE, EPV)
            SET TO  ← TO_CBF_GBC            # Eq. F.1 in clause F.3
        ELSE
            SET TO  ← TO_CBF_MAX
        ENDIF
        START TIMER(TO)
        RETURN 0                            # Indicates that packet is buffered
    ENDIF
ENDIF
IF (TIMER(TO) EXPIRES) THEN
    FETCH P FROM B
    SET NH_LL_ADDR ← BCAST
    RETURN P, NH_LL_ADDR
ENDIF
```

Annex G (normative): GeoNetworking traffic classification

GeoNetworking shall support traffic classification where each GeoNetworking packet is placed into a limited number of traffic classes. The traffic classification of GeoNetworking packets shall be based on the *TC* field in the GeoNetworking *Common Header* (clause 9.7.5).

GeoNetworking applies particular mechanisms for data traffic management to each traffic class differently. The GeoNetworking media-independent operations in the present document support SCF per traffic class. Further data traffic management mechanisms are specified in the media-dependent parts of GeoNetworking protocol specification.

A mapping between a traffic class to data traffic management mechanisms is configured by the GN management services at start-up (see annex K).

Annex H (normative): GeoNetworking protocol constants

The GeoNetworking protocol constants and their default/initial values shall be as specified in table H.1.

The protocol constants represent MIB attributes specified in annex I.

Table H.1: GeoNetworking protocol constants

Item	GeoNetworking protocol constant	Default/initial value	Comment
1	itsGnLocalGnAddr	1	GeoNetworking address of the GAGH of the GeoAdhoc router
2	itsGnLocalAddrConfMethod	MANAGED (1)	AUTO (0): Local GN_ADDR is configured from MIB MANAGED (1): Local GN_ADDR is configured via the GN management using the service primitive *CORE_GAGH* (annex K) ANONYMOUS (2): Local GN_ADDR is configured by the security entity
3	itsGnProtocolVersion	1	Version of the GeoNetworking protocol set in the GeoNetworking protocol headers
4	itsGnIsMobile	Stationary (0) Mobile (1)	Indicates whether ITS-S is stationary or mobile
5	itsGnIfType	Unspecified (0) ITS-G5 (1) LTE-V2X (2)	Indicates type of interface
6	itsGnMinUpdateFrequencyEPV	1 000	Minimum update frequency of EPV in [1/ms]
7	itsGnPaiInterval	80	Distance related to the confidence interval for latitude and longitude [m]. Used to determine the PAI (clause 9.5.2)
8	itsGnMaxSduSize	1 398	Maximum size of GN-SDU [octets] 1 500 - GN_MAX (88) - GNSEC_MAX (0)
9	itsGnMaxGeoNetworkingHeaderSize	88	GN_MAX: Maximum size of GeoNetworking header [octets] Without security, the size is determined by the *GeoUnicast* packet header as defined in clause 9.8.2. If the GeoNetworking packet is secured (clause 9.4) the maximum size of the GeoNetworking header is set to the size of the *Basic Header* and the size of the *Secured Packet*
10	itsGnLifetimeLocTE	20	Lifetime of location table entry [s]
11	itsGnSecurity	DISABLED (0) ENABLED (1)	Indicates whether GN security is enabled (1) or disabled (0)
12	itsGnSnDecapResultHandling	STRICT (0) NON-STRICT (1)	Indicates the handling of the SN-DECAP result code (service primitive *SN-ENCAP.confirm* parameter *report).* If the GN protocol constant `itsGnSnDecapResultHandling` is set to STRICT (0), received GN packets that are not correctly verified and decrypted (service primitive *SN-ENCAP.confirm* parameter *report* != SUCCESS) are always dropped. If `itsGnSnDecapResultHandling` is set to NON-STRICT (1), GN packets that are not correctly verified and decrypted can be passed to the upper protocol entity for further processing
13	itsGnLocationServiceMaxRetrans	10	Maximum number of retransmissions of LS Request packets

Item	GeoNetworking protocol constant	Default/initial value	Comment
14	itsGnLocationServiceRetransmitTimer	1 000	Duration of Location service retransmit timer [ms]
15	itsGnLocationServicePacketBufferSize	1 024	Size of Location service packet buffer [Octets]
16	itsGnBeaconServiceRetransmitTimer	3 000	Duration of Beacon service retransmit timer [ms]
17	itsGnBeaconServiceMaxJitter	`itsGnBeaconServiceRetransmitTimer` / 4	Maximum beacon jitter [ms]
18	itsGnDefaultHopLimit	10	Default hop limit indicating the maximum number of hops a packet travels
19	itsGnDPLLength	8	Length of Duplicate Packet List (DPL) per source (clause A.2)
20	itsGnMaxPacketLifetime	600	Upper limit of the maximum lifetime [s]
21	itsGnDefaultPacketLifetime	60	Default packet lifetime [s]
22	itsGnMaxPacketDataRate	100	Maximum packet data rate for a GAGH of a GeoAdhoc router [Ko/s]. If the mean (EMA) packet data rate a of a GAGH exceeds the value, packets from this GeoAdhoc router (source or sender) are not forwarded
23	itsGnMaxPacketDataRateEmaBeta	90	Weight factor for the Exponential Moving Average of the packet data rate PDR (clause B.2) in percent
24	itsGnMaxGeoAreaSize	10	Maximum size of the geographical area for a GBC and GAC packet [km^2]. If the geographical area size exceeds the maximum value, the GeoNetworking packet shall not be sent (source) and not be forwarded (forwarder)
25	itsGnMinPacketRepetitionInterval	100	Lower limit of the packet repetition interval [ms]
26	itsGnNonAreaForwardingAlgorithm	GREEDY (1)	Default forwarding algorithm outside target area
27	itsGnAreaForwardingAlgorithm	CBF (2)	Default forwarding algorithm inside target area
28	itsGnCbfMinTime	1	Minimum duration a GN packet shall be buffered in the *CBF packet buffer* [ms]
29	itsGnCbfMaxTime	100	Maximum duration a GN packet shall be buffered in the *CBF packet buffer* [ms]
30	itsGnDefaultMaxCommunicationRange	1 000	Default theoretical maximum communication range [m]
31	itsGnBroadcastCBFDefSectorAngle	30	Default threshold angle for advanced GeoBroadcast algorithm in clause F.4 [degrees]
32	itsGnUcForwardingPacketBufferSize	256	Size of *UC forwarding packet buffer* [Ko]
33	itsGnBcForwardingPacketBufferSize	1 024	Size of *BC forwarding packet buffer* [Ko]
34	itsGnCbfPacketBufferSize	256	Size of *CBF packet buffer* [Ko]
35	itsGnDefaultTrafficClass	0x00	Forwarding: Default traffic class

Annex I (informative):
ASN.1 specification of the GeoNetworking MIB

The ASN.1 specification of the GeoNetworking MIB is available at:

- https://forge.etsi.org/rep/ITS/asn1/gn_ts103836_4_1/-/blob/v2.1.1/ITSGN-MIB-3836-4-1_v0.0.4.mib

NOTE: Further information about dependencies and the development history is available at https://forge.etsi.org/rep/ITS/asn1/gn_ts103836_4_1https://forge.etsi.org/rep/ITS/asn1/gn_ts103836_4_1.

Annex J (informative): GeoNetworking data interface

J.1 General

The GN data service primitives allow entities of ITS transport protocols to send and receive packets via the TRANSP_CORE interface (figure 1).

J.2 *TRANSP_CORE.request*

The service primitive *TRANSP_CORE.request* is used by the ITS transport protocol entities to request sending a GN packet. Upon reception of the service primitive *TRANSP_CORE.request*, the GN Core delivers the transport packet to the active GAGH associated to the AliID provided via the CORE_GAGH interface.

The parameters of the *TRANSP_CORE.request* are as follows:

```
TRANSP_CORE.request (
            Upper protocol entity,
            Packet transport type,
            Destination address,
            AliID,
            Security profile, (optional)
            ITS-AID length, (optional)
            ITS-AID, (optional)
            Security permissions length, (optional)
            Security permissions, (optional)
            Security context information, (optional)
            Security target ID list length, (optional)
            Security target ID list,(optional)
            Maximum packet lifetime, (optional)
            Repetition interval, (optional)
            Maximum repetition time, (optional)
            Maximum hop limit, (optional)
            Traffic class,
            Length,
            Data
            )
```

The *Upper protocol entity* parameter specifies whether the service primitive was triggered by an ITS Transport protocol (e.g. BTP) or by the GeoNetworking to IPv6 Adaptation Sub-Layer (GN6ASL). In case the upper protocol entity is the IPv6 Adaptation Sub-Layer, the AliID for the message is requested to the MCO_NET entity.

The *Packet transport type* parameter specifies the packet transport type. The parameter can be either:

- (0) GUC;
- (1) SHB;
- (2) TSB;
- (3) GBC;
- (4) GAC.

The *Destination* parameter specifies the destination address for GeoUnicast or the geographical area for GBC/GAC. The destinations address for GeoUnicast can optionally contain the MID field only; with the other fields set to 0 (see figure 3 and table 1).

The *AliID* parameter indicates the ALI at the Access layer that should be used for transmission and the GAGH to use to handle this packet.

The *Security profile* parameter determines the security service to invoke.

The *ITS-AID length* parameter specifies the length of the value provided in the *ITS-AID* parameter.

The *ITS-AID* parameter specifies the ITS-AID for the payload to be sent.

The *Security permissions length* parameter specifies the length of the value provided in the *Security permissions* parameter.

The *Security permissions* parameter specifies the SSP associated with the ITS-AID.

The *Security context information* parameter specifies information to be used to selecting properties of the security protocol.

The *Security target ID list length* parameter specifies the length for the value of the *SecurityTarget ID List* parameter.

The *Security target ID list* parameter specifies an unordered collection of target IDs used by the security entity, for specifying multiple recipients.

The *Maximum lifetime* parameter specifies the maximum tolerable time in [s] a GeoNetworking packet can be buffered until it reaches its destination. The parameter is optional. If it is not used, the GN protocol constant `itsGnDefaultPacketLifetime` is used.

The *Repetition interval* parameter specifies the duration between two consecutive transmissions of the same GeoNetworking packet during maximum repetition time of a packet in [ms]. The parameter is optional. If it is not used, the packet is not repeated.

The *Maximum repetition time* parameter specifies the duration in [ms] for which the packet will be repeated if the Repetition interval is set. The parameter is optional; if the Repetition interval is not used, it is omitted.

The *Maximum Hop Limit* specifies the number of hops a packet is allowed to have in the network, i.e. how often the packet is allowed to be forwarded.

The *Traffic class* parameter specifies the traffic class for the message.

The *Length* parameter indicates the length of the *Data*.

The *Data* parameter represents the payload of the GeoNetworking packet to be sent.

J.3 *TRANSP_CORE.confirm*

The service primitive *TRANSP_CORE.confirm* is used to confirm that the GeoNetworking packet was successfully processed in response to a *TRANSP_CORE.request*. For the reception of the primitive, no behaviour is specified.

The parameters of the service primitive are as follows:

```
TRANSP_CORE.confirm (
            ResultCode
            )
```

The *ResultCode* parameter specifies whether the service primitive *TRANSP_CORE.request* is:

1) accepted;

2) rejected due to maximum length exceeded if the size of the T/GN6-PDU exceeds the GN protocol constant `itsGnMaxSduSize`;

3) rejected due to maximum lifetime exceeded if the lifetime exceeds the maximum value of the GN protocol constant `itsGnMaxPacketLifetime`;

4) rejected due to repetition interval too small, if the repetition interval is smaller than the GN protocol constant `itsGnMinPacketRepetitionInterval`;

5) rejected due to unsupported traffic class;

6) rejected due to non-valid AliID;

7) rejected due to geographical area exceeds the maximum geographical area size in the GN protocol constant `itsGnMaxGeoAreaSize`; or

8) rejected for unspecified reasons if the service primitive *TRANSP_CORE.request* cannot be accepted for any other reason.

J.4 *TRANSP_CORE.indication*

The service primitive *TRANSP_CORE.indication* indicates to an upper protocol entity that a GeoNetworking packet has been received. The service primitive is generated by the GN Core to deliver data contained in a received GeoNetworking packet to upper protocol entity. The data of the GeoNetworking packet are processed as determined by the receiving upper protocol entity.

The parameters of the service primitive *TRANSP_CORE.indication* are as follows:

```
TRANSP_CORE.indication (
            Upper protocol entity,
            Packet transport type,
            AliID,
            Destination, (optional)
            Source position vector,
            Security report, (optional)
            Certificate id, (optional)
            ITS-AID length, (optional)
            ITS-AID, (optional)
            Security permissions length, (optional)
            Security permissions, (optional)
            Traffic class,
            Remaining packet lifetime, (optional),
            Remaining hop limit, (optional)
            Length,
            Data    -- T/GN6-PDU
            )
```

The *Upper protocol entity* parameter determines the protocol entity that processes the service primitive (BTP or GN6).

The *Packet transport type* parameter is the packet transport type (GUC, SHB, TSB, GBC, GAC) of the received packet.

The *AliID* parameter indicates the ALI at the Access layer that received the packet.

The *Destination* parameter is the destination address for GeoUnicast or the geographical area for GeoBroadcast/GeoAnycast with which the GeoNetworking packet was generated by the source.

The *Source position vector* parameter is the geographical position for the source of the received GeoNetworking packet.

The *Security report* contains result information from the security operations for decryption and verification (parameter *report* in the service primitive *SN-DECAP.confirm*).

The *Certificate id* contains the identification of source certificate, for example the certificate hash (parameter *certificate_id* in the service primitive *SN-DECAP.confirm*).

The *ITS-AID length* parameter specifies the length of the value provided in the *ITS-AID* parameter (parameter *its_aid_length* in the service primitive *SN-DECAP.confirm*).

The *ITS-AID* parameter specifies the ITS-AID for the received payload (parameter *its_aid* in the service primitive *SN-DECAP.confirm*).

The *Security permissions length* parameter specifies the length of the value provided in the *Security permissions* parameter (parameter *permissions_length* in the service primitive *SN-DECAP.confirm*).

The *Security permissions* parameter contains the sender permissions (parameter *permissions* in the service primitive *SN-DECAP.confirm*).

The *Traffic class* parameter is the traffic class, with which the GeoNetworking packet was generated by the source.

The *Remaining packet lifetime* parameter is the remaining lifetime of the packet.

The *Remaining hop limit* parameter is the remaining hop limit of the packet.

The *Length* parameter is the length of the *Data* parameter.

The *Data* parameter is the payload of the received GeoNetworking packet, i.e. the T-PDU/GN6-PDU.

Annex K (informative): GN Core network management services

K.1 General

The GN network management service primitives provide position, time and GeoNetworking address updates to the GAGHs.

K.2 *CORE_GAGH.request*

The service primitive *CORE_GAGH.request* is generated by the GeoNetworking protocol upon activation of a GAGH in order to request management information, i.e. time, position vector, GeoNetworking address, TC mapping. After receiving the service primitive *CORE_GAGH.request*, the GN Core is in charge of providing the GAGH with the requested management information.

The parameter of the *CORE_GAGH.request* is as follows:

```
CORE_GAGH.request (
            Request cause
            )
```

The *Request cause* parameter specifies the type of requested information, i.e. time, position vector, GeoNetworking address, TC mapping. In case the GeoNetworking address is requested, the parameter also indicates whether the address request is caused by DAD or is an initial request.

K.3 *CORE_GAGH.response*

The service primitive *CORE_GAGH.response* is generated by the GN Core to indicate an update of management information, i.e. time, position vector, GeoNetworking address and TC mapping. The service primitive can be triggered upon reception of a *CORE_GAGH.request* primitive or can be generated unsolicited, i.e. without a service primitive *CORE_GAGH.request*.

The parameters of the *CORE_GAGH.response* are as follows:

```
CORE_GAGH.response (
                Time (optional)
                Local position vector (optional)
                GeoNetworking address (optional)
                TC mapping (optional)
                )
```

The *Time* parameter specifies the timestamp that is used as a reference to determine the freshness of received information carried in packets.

The *Local position vector* parameter specifies the ITS-S's most recent position vector (geographical position, speed, heading, timestamp when the position vector was generated, and corresponding accuracy information).

The *GeoNetworking address* parameter specifies the GeoNetworking address that is used by the GeoNetworking protocol.

The *TC mapping* parameter specifies the mapping of Traffic class IDs to TC-related parameters (annex G).

All parameters are optional, whereas at least one parameter is present.

K.4 *CORE_GAGH.indication*

The service primitive *CORE_GAGH.indication* indicates to the GN Core entity that a GeoNetworking packet has been received. The service primitive is generated by a GAGH to deliver data contained in a received GeoNetworking packet to upper protocol entity. The data of the GeoNetworking packet are processed as determined by the receiving upper protocol entity.

The parameters of the service primitive *CORE_GAGH.indication* are as follows:

```
TRANSP_CORE.indication (
            Upper protocol entity,
            Packet transport type,
            AliID,
            Destination, (optional)
            Source position vector,
            Security report, (optional)
            Certificate id, (optional)
            ITS-AID length, (optional)
            ITS-AID, (optional)
            Security permissions length, (optional)
            Security permissions, (optional)
            Traffic class,
            Remaining packet lifetime, (optional),
            Remaining hop limit, (optional)
            Length,
            Data    -- T/GN6-PDU
            )
```

The *Upper protocol entity* parameter determines the protocol entity that processes the service primitive (BTP or GN6).

The *Packet transport type* parameter is the packet transport type (GUC, SHB, TSB, GBC, GAC) of the received packet.

The *AliID* parameter indicates the ALI at the Access layer that received the packet.

The *Destination* parameter is the destination address for GeoUnicast or the geographical area for GeoBroadcast/GeoAnycast with which the GeoNetworking packet was generated by the source.

The *Source position vector* parameter is the geographical position for the source of the received GeoNetworking packet.

The *Security report* contains result information from the security operations for decryption and verification (parameter *report* in the service primitive *SN-DECAP.confirm*).

The *Certificate id* contains the identification of source certificate, for example the certificate hash (parameter *certificate_id* in the service primitive *SN-DECAP.confirm*).

The *ITS-AID length* parameter specifies the length of the value provided in the *ITS-AID* parameter (parameter *its_aid_length* in the service primitive *SN-DECAP.confirm*).

The *ITS-AID* parameter specifies the ITS-AID for the received payload (parameter *its_aid* in the service primitive *SN-DECAP.confirm*).

The *Security permissions length* parameter specifies the length of the value provided in the *Security permissions* parameter (parameter *permissions_length* in the service primitive *SN-DECAP.confirm*).

The *Security permissions* parameter contains the sender permissions (parameter *permissions* in the service primitive *SN-DECAP.confirm*).

The *Traffic class* parameter is the traffic class, with which the GeoNetworking packet was generated by the source.

The *Remaining packet lifetime* parameter is the remaining lifetime of the packet.

The *Remaining hop limit* parameter is the remaining hop limit of the packet.

The *Length* parameter is the length of the *Data* parameter.

The *Data* parameter is the payload of the received GeoNetworking packet, i.e. the T-PDU/GN6-PDU.

Annex L (informative): MCO_NET services for the GN Core

L.1 General

The MCO_NET entity provides services to the GN Core to provide GAGH status updates and to request ALI for packets without assigned AliID.

L.2 *CORE_MCO.request*

The service primitive *CORE_MCO.request* is generated by the GN Core upon reception of a packet request without assigned AliID, e.g. in the case of transmission request from the IPv6 Adaptation Sub-Layer.

The parameter of the *CORE_MCO.request* is as follows:

```
CORE_GAGH.request (
            ALI request
            Request number
            )
```

The *ALI request* parameter identifies the ALI request type. The request type can have the following type:

- (0) IPv6 ALI request;
- (1) Other.

The *Request number* parameter identifies the request that will be used by MCO_NET for the response.

L.3 *CORE_MCO.response*

The service primitive *CORE_MCO.response* is generated by MCO_NET to indicate a response of a previously performed CORE_MCO.request.

```
CORE_MCO.response (
                ALI request
                Request number
                AliID
                Expiration Time
                )
```

The *ALI request* parameter identifies the ALI request type. The type of the request can be:

- (0) IPv6 ALI request; or
- (1) Other.

The *Request number* parameter is the same as the one indicated in the CORE_MCO.request.

The *AliID* parameter indicates the ALI that should be used for the packet type as described in the ALI request.

The *Expiration Time* is the time until which the provided AliID is valid for the performed request.

L.4 *CORE_MCO.indication*

The service primitive *CORE_MCO.indication* is generated by MCO_NET to indicate the status update of a GAGH to the GN Core.

The parameters of the *CORE_MCO.indication* are as follows:

```
CORE_MCO.indication (
                GAGH
                GAGH Status
                )
```

The *GAGH* parameter indicates the GAGH for which the status update is.

The *GAGH Status* parameter contains the new status for the GAGH indicated. The status can be (1) Active, (2) Inactive, or (3) Disabled.

Annex M (informative): MCO_NET services for the GAGHs

M.1 General

The MCO_NET entity provides services to the GAGHs to receive and collect network updates and status updates.

M.2 *GAGH_MCO.request*

The service primitive *GAGH_MCO.request* is generated by a GAGH to provide a network status update to MCO_NET.

The parameter of the *GAGH_MCO.request* is as follows:

```
CORE_GAGH.requestrequest (
            GAGH ID
            Network Update
            Access Technology
            )
```

The *GAGH ID* parameter is the ID of the GAGH indicating its network update.

The *Network Update* parameter contains the parameter values of the network update. These parameters may depend on the access technology handled by the GAGH.

The *Access Technology* identifies the access technology handled by the GAGH.

M.3 *GAGH_MCO.indication*

The service primitive *GAGH_MCO.indication* is generated by MCO_NET to indicate a received status update or a local CLR from the Access layer to a GAGH.

The parameter of the *GAGH_MCO.indicate* is as follows:

```
GAGH_MCO.indication (
                State (optional)
                CLR (optional)
                )
```

The optional *State* parameter indicates the new state of the GAGH as described in clause 10.4.2.2.

The optional *CLR* parameter indicates the Channel Load Ratio perceived for an ALI group.

Annex N (informative): MCO services for the Facilities layer

N.1 General

The MCO_NET data service primitives for the Facilities layer enable entities of the Facilities layer to request MCO parameter configurations, receive MCO parameter and network updates, and handle ALI requests with the MCO_FAC_NET interface (figure 1).

N.2 MCO_FAC_NET.*request*

The service primitive *MCO_FAC_NET.request* is generated by a Facilities layer entity to perform a request for MCO parameters configuration to MCO_NET.

The parameters of the *MCO_FAC_NET.request* is as follows:

```
CORE_GAGH.request (
            AliGroupID
            AliGroupID status
            )
```

The *AliGroupID* is the ALI group for which the parameters are provided.

The *AliGroupID status* parameter contains the new requested status for the AliGroupID indicated. The status can be either:

- (0) deactivation;
- (1) activate message reception;
- (2) active message transmission and reception.

N.3 MCO_FAC_NET.*indication*

The service primitive *MCO_FAC_NET.indication* is generated by MCO_NET when:

1) the access layer provided an update of an ALI group status; or
2) the access layer provided an access layer notification; or
3) GN Core performed an ALI request; or
4) MCO_NET performed a network update.

The parameters of the *MCO_FAC_NET.indication* are as follows:

```
MCO_FAC_NET.indication  (
                AliGroupID (optional)
                Network status (optional)
                Access layer notification (optional)
                ALI request (optional)
                Request number (optional)
                )
```

The *AliGroupID* parameter is the ALI group identifier of the ALI group for which the update is provided. This parameter is only optional in the case the update is about an ALI request.

The *Network status* parameter contains information about the network status for the indicated AliGroupID. It may for example contain the local CLR or the CLRs received by other ITS-Ss.

The *Access layer notification* parameter is provided by the Access layer entities and forwarded to the Facilities layer via this interface.

An update is generated by MCO_NET if one of the following conditions is met:

1) The Access layer provided an update of an ALI group state. In this case, the *AliGroupID* and *Network status* parameters are required. The *Network status* should contain the new status of the indicated ALI group.

2) The Access layer provided an Access layer notification to MCO_NET. In this case, the parameters *AliGroupID* and *Access layer notification* are required.

3) An ALI is requested by the GN_CORE entity to MCO_NET. In this case, the parameters *ALI request* and *Request number* are required.

4) Depending on the required information to operate at the Facilities layer, periodic updates may be generated with the parameters *AliGroupID* and *Network status.* The parameter *Network status* contains the relevant information for the Facilities layer, e.g. the locally perceived CLR or the CLRs received by other ITS-Ss for the indicated AliGroupID.

N.4 MCO_FAC_NET.*response*

The service primitive *MCO_FAC_NET.response* is generated by the Facilities layer entities to the MCO_NET entity to indicate the selected AliID for an ALI request indicated by MCO_NET using the MCO_NET.indicate primitive.

```
MCO_FAC_NET.response    (
                ALI request
                Request number
                AliID
                Expiration time
                )
```

The *ALI request* should be the same as indicated in MCO_FAC_NET.indicate service primitive used to indicate the AliID request.

The *Request number* parameter should have the same value as the initial request performed by the MCO_NET entity.

The *AliID* parameter is the selected *AliID* for the performed ALI request.

The *Expiration time* parameter is the period of time for which the selected *AliID* can be used for the ALI request.

Annex O (informative): Bibliography

- ETSI TS 102 890-1: "Intelligent Transport Systems (ITS); Facilities layer function; Part 1: Services Announcement (SA) specification".
- ETSI TS 102 890-2: "Intelligent Transport Systems (ITS); Facilities layer function; Part 2: Position and time facility specification".
- ETSI TR 102 707: "Intelligent Transport Systems (ITS); ETSI object identifier tree; ITS domain".
- ETSI TS 102 723-1: "Intelligent Transport Systems; OSI Cross-Layer Topics; Part 1: Architecture and Addressing Schemes".
- EU FP7 GEONET Project: "Deliverable D2.2 Final GeoNet Specification", ETSI Document # ITSWG3(10)0011, January 2010.
- SIM TD Project: "Deliverable D21.4 Spezifikation der Kommunikationsprotokolle", September 2009.
- ISO/TS 1824-2:2006: "Traffic and Travel Information (TTI) -- TTI via Transport Protocol Expert Group (TPEG) data-streams -- Part 2: Syntax, Semantics and Framing Structure (SSF)".
- M. Torrent Moreno, J. Mittag, P. Santi, H. Hartenstein: "Vehicle-to-Vehicle Communication: Fair Transmit Power Control for Safety-Critical Information", IEEE Transactions on Vehicular Technology, Volume 58, Issue 7, pp. 3684-3707, September 2009.
- A. Festag, P. Papadimitratos, T. Tielert: "Design and Performance of Secure Geocast for Vehicular Communication", IEEE Transactions on Vehicular Technology, Volume 59, Issue 5, pp. 1 - 16, June 2010.
- M. Wetterwald, F. Hrizi, P. Cataldi: "Cross-layer identities management in ITS stations", 10th IEEE International Conference on ITS Telecommunications (ITST), November 2010, Kyoto, Japan.
- M.N. Mariyasagayam, T. Osafune, M. Lenardi: "Enhanced Multi-Hop Vehicular Broadcast (MHVB) for Active Safety Applications", 7th IEEE International Conference on ITS Telecommunications (ITST), June 2007.
- ETSI EN 302 663: "Intelligent Transport Systems (ITS); ITS-G5 Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band".
- ISO/IEC 8802-2:1998: "Information technology -- Telecommunications and information exchange between systems -- Local and metropolitan area networks -- Specific requirements -- Part 2: Logical link control".

Annex P (informative): Change History

Date	Version	Information about changes
2022-01	0.0.2	Created base version of draft based on ETSI EN 302 636-4-1 v1.4.1 GeoNetworking media-independent functionality (Release 1) and converted from EN template to TS template
2022-02	0.0.3	Changes based on ITSWG3(22)057004r1 "Proposed MCO-related extensions to GeoNetworking media-independent Release 2 (rev 1)"
2022-05	0.0.4	Clean up references, add informative annexes L, M, and N and modify Clause 10 figures and text
2022-06	0.0.5	Comment resolution: change clause N.3 and fix typos
2022-06	0.0.6	Comment resolution
2022-07	0.0.7	Clean version
2022-09	0.0.8	Comment resolution after TC approval
2022-09	0.0.9	Cleanup for approval

History

Document history		
V2.1.1	November 2022	Publication