Draft ETSI EN 302 636-4-1 V1.2.0 (2013-10)

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

Reference
REN/ITS-0030035

Keywords
autonomic networking, ITS, network, safety

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 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is 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 (http://ipr.etsi.org).

Pursuant to the ETSI IPR Policy, no investigation, 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.

Foreword

This draft European Standard (EN) has been produced by ETSI Technical Committee Intelligent Transport Systems (ITS), and is now submitted for the combined Public Enquiry and Vote phase of the ETSI standards EN Approval Procedure.

The present document is part 4, sub-part 1 of a multi-part deliverable. Full details of the entire series can be found in part 1 [2].

Proposed national transposition dates	
Date of latest announcement of this EN (doa):	3 months after ETSI publication
Date of latest publication of new National Standard or endorsement of this EN (dop/e):	6 months after doa
Date of withdrawal of any conflicting National Standard (dow):	18 months after doa

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 media-independent functionality of the GeoNetworking protocol.

2 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 http://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.

2.1 Normative references

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

[1] ETSI EN 302 665: "Intelligent Transport Systems (ITS); Communications Architecture".

[2] ETSI EN 302 636-1: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 1: Requirements".

[3] ETSI EN 302 636-2: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 2: Scenarios".

[4] ETSI TS 102 636-3: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 3: Network architecture".

[5] ETSI EN 302 636-5-1: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 5: Transport Protocols; Sub-part 1: Basic Transport Protocol".

[6] ETSI EN 302 636-6-1: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 6: Internet Integration; Sub-part 1: Transmission of IPv6 Packets over GeoNetworking Protocols".

[7] ETSI EN 302 931: "Intelligent Transport Systems (ITS); Vehicular Communications; Geographical Area Definition".

[8] Annex to ITU Operational Bulletin No. 741 - 1.VI.2001: "Complement to Recommendation ITU-T E.212 (11/98)".

NOTE: Available at: http://www.itu.int/ITUT/.

2.2 Informative references

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 EN 302 663: "Intelligent Transport Systems (ITS); Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band".

[i.2] ETSI TS 102 636-4-2: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 2: Media dependent functionalities for ITS-G5A media".

[i.3] ETSI TS 102 731: "Intelligent Transport Systems (ITS); Security; Security Services and Architecture".

[i.4] ETSI TS 102 723-8: "Intelligent Transport Systems (ITS); OSI cross-layer topics; Part 8: Interface between security entity and network and transport layer".

[i.5] ETSI TS 103 097: "Intelligent Transport Systems (ITS); Security; Security header and certificate formats".

[i.6] ETSI TS 102 890-3: "Intelligent Transport Systems (ITS); Facilities layer function; Facility Position and time management".

[i.7] ETSI TS 102 894-2: "Intelligent Transport Systems (ITS); Users and applications requirements; Part 2: Applications and facilities layer common data dictionary".

[i.8] 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".

[i.9] IETF RFC 2578: "Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)". Textual Conventions for SMIv2.

[i.10] 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.11] IETF RFC 2579: "Textual Conventions for SMIv2".

[i.12] 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.13] ETSI EN 302 636-4: "Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications".

3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in EN 302 665 [1], TS 102 636-3 [4], EN 302 636-6-1 [6] and the following apply:

destination: receiver that processes a GUC 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: ad hoc router that implements the GeoNetworking protocol

local position vector: position vector for the local GeoAdhoc router

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 and relays the packet to other GeoAdhoc routers

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 following symbols apply:

GEO_MAX	Maximum size of the GeoNetworking packet header
H(GN_ADDR)	Heading of the ITS-S GN_ADDR
LAT	Latitude
LL_ADDR	Link layer address that identifies the ITS-S at the link layer protocol entity in the ITS Access Layer
LL_ADDR_NH	Link layer address of the next hop
LONG	Longitude
LS_PENDING	Location Service pending flag
MTU_AL	MTU of the ITS 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
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(GN_ADDR)	Last maximum sequence number received from a GeoAdhoc router
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 EN 302 665 [1], TS 102 636-3 [4], EN 302 636-6-1 [6] and the following apply:

ASN	Abstract Syntax Notation
BC	BroadCast
BTP	Basic Transport Protocol
CBF	Contention-Based Forwarding
DE	DEstination
EMA	Exponential Moving Average
FCS	Frame Check Sequence
FIFO	First In First Out
GAC	Geographically-Scoped Anycast
GBC	Geographically-Scoped Broadcast
GF	Greedy Forwarding
GN	GeoNetworking
GN_ADDR	GeoNetworking ADDRess
GN6ASL	GeoNetworking to IPv6 Adaptation Sub-Layer
GN6-SDU	GN6 Service Data Unit
GN-PDU	GeoNetworking Protocol Data Unit
GN-SDU	GeoNetworking Service Data Unit
GUC	Geographically-Scoped Unicast

HST	Header Sub-Type
HT	Header Type
LL	Link Layer
LLC	Logic Link Control
LocT	Location Table
LocTE	Location Table Entry
LPV	Local Position Vector
LS	Location Service
LT	LifeTime
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
MTU	Maximum Transmit Unit
NH	Next Header
PAI	Position Accuracy Indicator
PCI	Protocol Control Information
PDR	Packet Data Rate
PDU	Protocol Data Unit
PL	Payload Length
POS	POSition
PV	Position Vector
RHL	Remaining Hop Limit
RTC	Retransmit Counter
SCC	Station Country Code
SCF	Store Carry & Forward
SDU	Service Data Unit
SE	SEnder
SHB	Single Hop Broadcast
SN	Sequence Number
SO	SOurce
SPV	Short Position Vector
ST	Station Type
TAI	Temps Atomique International (International Atomic Time)
TC ID	Traffic Class Identifier
TC	Traffic Class
TSB	Topologically Scoped Broadcast
T-SDU	Transport Service Data Unit
TST	TimeSTamp
UC	UniCast
UTC	Universal Time Coordinated
WGS	World Geodetic System

4 Services provided by the GeoNetworking protocol

The GeoNetworking protocol is a network protocol that resides in the ITS network and transport layer [1] and is executed in the ad hoc router [4], specifically in the GeoAdhoc router. It provides the transport of packets in the ITS ad hoc network [4]. It shall support the requirements specified in [2] and the scenarios specified in [3].

The GeoNetworking protocol provides services to upper protocol entities, i.e. the ITS Transport Protocol, such as the Basic Transport Protocol (BTP) [5], and the GeoNetworking to IPv6 Adaptation Sub-Layer (GN6ASL) [6]. The services are provided via the GN_SAP using service primitives of different types that carry parameters and the PDU of the upper protocol entity, i.e. T/GN6 PDU (see figure 1). A PDU of the transport protocols is considered as SDU in the GeoNetworking protocol. The SDU is complemented with Protocol Control Information (PCI) and transmitted as GN PDU to the peer entity.

In order to provide its packet transport services, the GeoNetworking protocol uses the services of the ITS Access Layer.

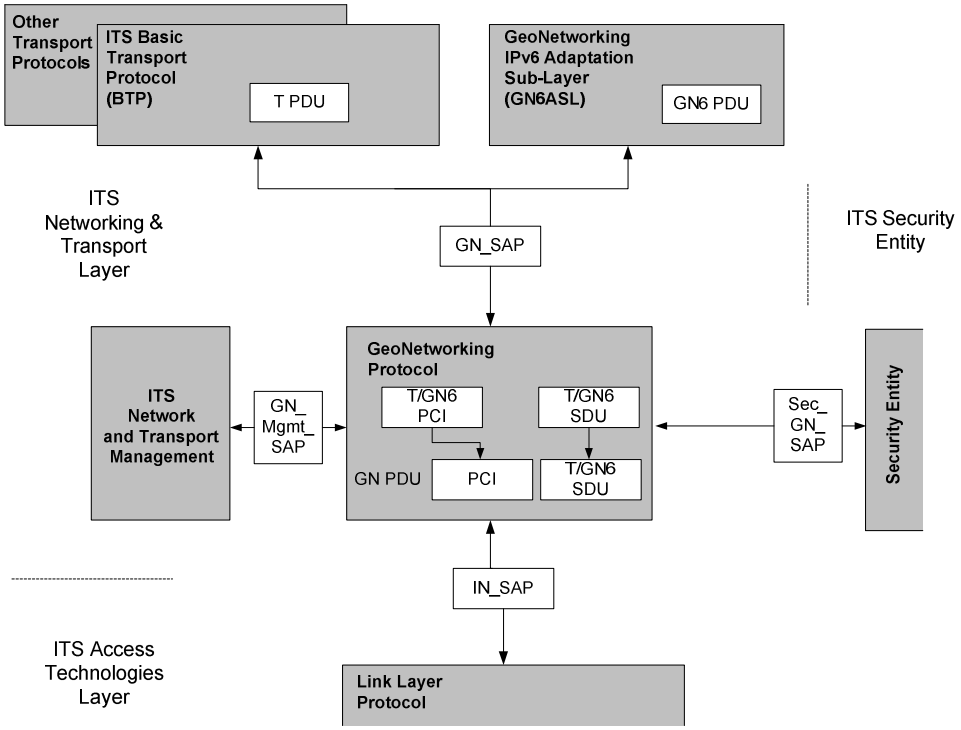

Figure 1: Service primitives, SDUs and PDUs relevant for the GeoNetworking protocol

Figure 1 illustrates the interfaces and SAPs of the ITS networking and transport layer as specified in [4]. The present document specifies the internal GN_SAP between the GeoNetworking protocol and the ITS transport protocol, such as the Basic Transport Protocol (BTP) [5], the GeoNetworking IPv6 Adaptation Sub-Layer (GN6ASL) as defined in [6] and other transport protocols, the GN_Mgmt_SAP between the GeoNetworking protocol and the *ITS Networking & Transport Layer Management*, as well as the Sec_GN_SAP between the GeoNetworking protocol and the ITS Security.

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.

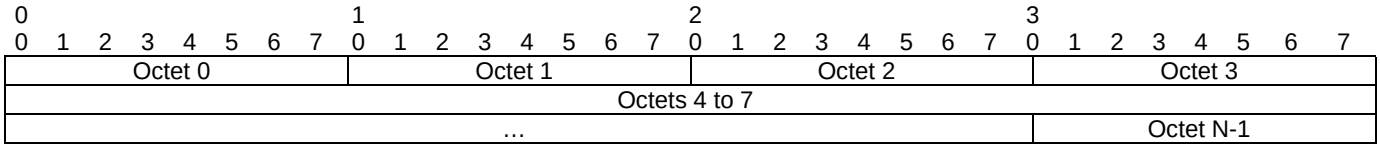

Figure 2: Format convention

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 should 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 9.2.1.5 is applied.

6.2 GeoNetworking address format

The format of the GeoNetworking address is described in figure 3.

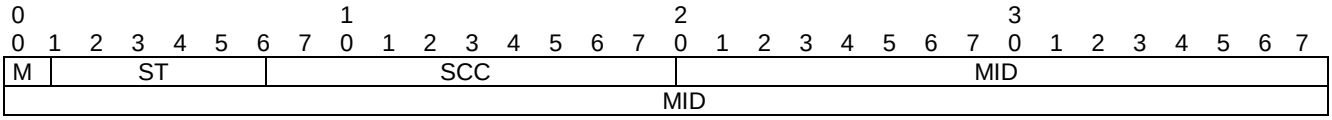

Figure 3: GeoNetworking address format

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 9.2.1.3.2) (update) and the initial GeoNetworking address (9.2.1.3.1). M is set to 1 if the address is manually configured otherwise it equals 0.
2	ST	Octet 0 Bit 1	Octet 0 Bit 5	5 bit unsigned integer	ITS-S type. To identify the ITS-S type. 0 - Unknown 1 - Pedestrian 2 - Cyclist 3 - Moped 4 - Motorcycle 5 - Passenger Car 6 - Bus 7 - Light Truck 8 - Heavy Truck 9 - Trailer 10 - Special Vehicle 11 - Tram 15 - Road Side Unit (see note)
4	SCC	Octet 0 Bit 6	Octet 1 Bit 7	10 bit unsigned integer	ITS-S Country Code.
5	MID	Octet 2	Octet 7	48 bit address	This field represents the LL_ADDR.
NOTE: The values of the ITS-S type are aligned with [i.7] "Applications and facilities layer common data dictionary".					

The first bit is reserved for the recognition of manual configured GeoNetworking addresses. The allocation of ITS-S Country Codes (SCC) is administered by the ITU-T [8].

The MID field corresponds to the access layer address. In case of ITS-G5 [i.1] MAC layer, the 48-bit MAC layer address shall be used.

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.12], the GeoNetworking address space shall remain 48-bit wide (size of the MID field in the GeoNetworking address). In particular, as described in EN 302 636-6-1 [6], table 1, note 1, the GN6ASL resolves an MID from a unicast destination IPv6 address and passes it to GeoNetworking via the service primitive *GN-DATA.request* (clause I.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 layer and to perform the forwarding via GeoNetworking in a transparent way (see EN 302 636-6-1 [6]).

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 (pseudonym [i.3]).

7 Data structures

7.1 Location table

7.1.1 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 that execute the GeoNetworking protocol. The data elements of a location table entry are specified in clause 7.1.2 and the maintenance of the location table in clause 7.1.3.

7.1.2 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*.
- LL address of the ITS-S *LL_ADDR*.
- Type of the ITS-S (e.g. vehicle ITS-S, roadside ITS-S).
- Version of the GeoNetworking used by the ITS-S.
- Position vector *PV*, i.e. Long Position Vector *LPV* (clause 8.5.2), 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 9.2.4) is in progress.
- Flag *IS_NEIGHBOUR(GN_ADDR)*: Flag indicating that the GeoAdhoc router is in direct communication range, i.e. is a neighbour.
- Sequence number *SN(GN_ADDR*): The sequence number of the last packet from the source *GN_ADDR* that was identified as 'not duplicated'.

- 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 can 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.

7.1.3 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 expires.

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

7.2 Local Position Vector

7.2.1 General

A GeoAdhoc router shall maintain a local data structure that holds position-related information for the local GeoAdhoc router, i.e. the local position vector LPV. The data elements of a local position vector are specified in clause 7.2.2 and the maintenance of the location table in clause 7.2.3.

7.2.2 Minimum data elements

The LPV shall contain at least the following data elements:

1) Geographical position *POS_LPV*.
2) Speed *S_LPV*.
3) Heading *H_LPV*.
4) Timestamp *TST_LPV* indicating when the geographical position *POS_LPV* was generated.
5) Accuracy of the geographical position *PAI*.

7.2.3 Maintenance

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

The LPV should be updated with a frequency of the GN protocol constant `itsGnMinUpdateFrequencyLPV` or higher.

NOTE: The GN protocol constant `itsGnMinUpdateFrequencyLPV` should have different values for vehicle ITS-S and roadside ITS-S. This reflects the fact that roadside ITS-S are not mobile, but at most portable, and therefore require a smaller update frequency.

7.3 Sequence number

7.3.1 General

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

7.3.2 Maintenance

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.

7.4 Location service packet buffer

7.4.1 General

Upon invocation of the LS (clause 9.2.4), 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 9.2.4).

7.4.2 Maintenance

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

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, the *LT* field shall be reduced by the queuing time in the *LS packet buffer*.

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

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

7.5 Forwarding packet buffer

7.5.1 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.

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

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

1) the GN protocol constant `itsGnGeoUnicastForwardingAlgorithm` is set to 2 (CBF),

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

7.5.2 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`.

7.5.3 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, the value of the *LT* field shall be reduced by the queuing time in the forwarding packet buffer.

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.

4) When a stored GeoNetworking packet is sent, the value of the *LT* field shall be reduced by the queuing time in the *CBF packet buffer*.

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

8 GeoNetworking packet structure and formats

8.1 Overview

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

8.2 Packet structure

8.2.1 General

As specified in TS 102 636-3 [4], the GeoNetworking protocol shall either be used in the GeoNetworking protocol stack (TS 102 636-3 [4], clause 7.3.2) or in the protocol stack that combines the GeoNetworking protocol and IPv6 (TS 102 636-3 [4], clause 7.3.4).

8.2.2 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 *MAC header* is the header of the MAC protocol of the ITS access technology. The MAC protocol can add additional protocol elements, such as a trailer for the MAC FCS as in ITS-G5 [i.1].

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

2) The LLC header is the header of 802.2 LLC/SNAP specified in ISO/IEC 8802-2 [i.8] with the Ethernet Type Field 0x8947 indicating GeoNetworking as the LLC transport protocol.

3) The *GeoNetworking header* is the header of the GeoNetworking packet as defined in the present document and extended for media-dependent GeoNetworking functionality, such as for ITS-G5 specified in TS 102 636-4-2 [i.2].

4) 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: The general packet structure is shown as seen by the MAC protocol of the ITS Access Layer.

NOTE 3: Some GeoNetworking packets do not carry a payload, such as Beacon.

MAC Header	LLC Header	GeoNetworking Header	Payload (optional)

Figure 4: GeoNetworking packet structure (without security)

8.2.3 Maximum Transmit Unit

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

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

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

8.3 GeoNetworking header structure

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

Figure 5: GeoNetworking header structure

Basic Header, *Common Header* and *Extended Header* are specified in clauses 8.6, 8.7, and 8.8.

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

8.4 GeoNetworking Secured Packet

The overall packet structure may be protected by security services [i.4], [i.5], i.e. by digital signatures and certificates and by encryption.

With enabled security (GN protocol constant `itsGnSecurity`), the overall packet structure is depicted in figure 6.

Security operations are executed by the security entity via the SAP Sec_GN_SAP (figure 1) and as specified in clause 9.3.

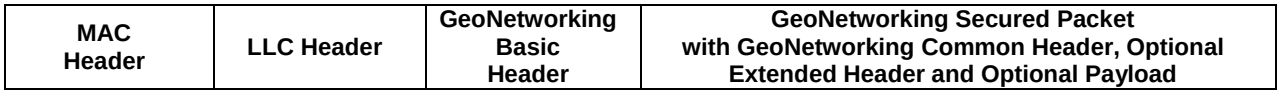

Figure 6: GeoNetworking packet structure (with security)

8.5 Position vector

8.5.1 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.

2) Short position vector.

8.5.2 Long Position Vector

8.5.2.1 Structure

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

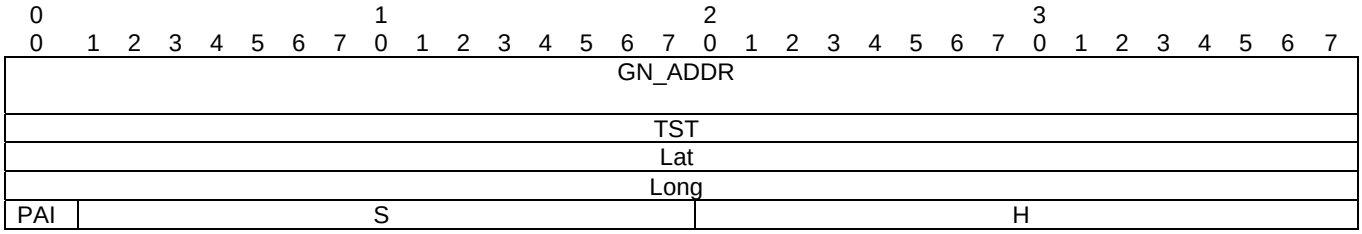

Figure 7: *Long Position Vector*

8.5.2.2 Fields

The *Long Position Vector* 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	The 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.10] latitude and longitude of the GeoAdhoc router reference position as specified in [i.6] 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 if the semiMajorConfidence of the PosConfidenceEllipse as specified in TS 102 894-2 [i.7] is smaller than the GN protocol constant `itsGnPaiInterval` / 2. Set to 0 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.

8.5.3 Short Position Vector

8.5.3.1 Structure

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

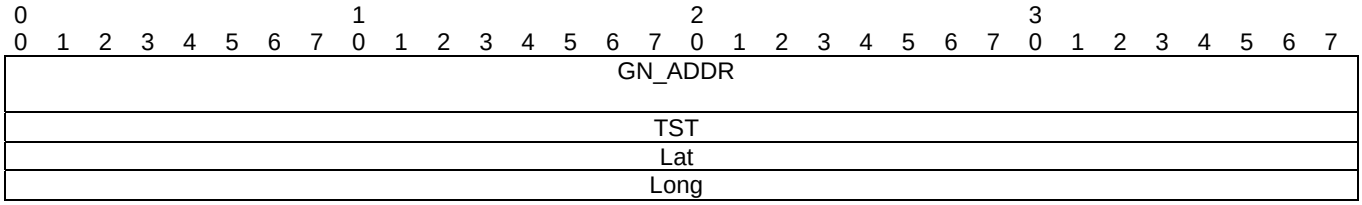

Figure 8: *Short Position Vector*

8.5.3.2 Fields

The *Short Position Vector* 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	The *GN_ADDR* field as specified in table 2.
2	*TST*	Octet 8	Octet 11	32 bit unsigned integer	[ms]	The Timestamp TST field as specified in table 2.
3	*Lat*	Octet 12	Octet 15	32 bit signed integer	[1/10 micro-degree]	The Latitude (*Lat*) field as specified in table 2.
4	*Long*	Octet 16	Octet 19	32 bit signed integer	[1/10 micro-degree]	The 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.

8.6 Basic Header

8.6.1 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.

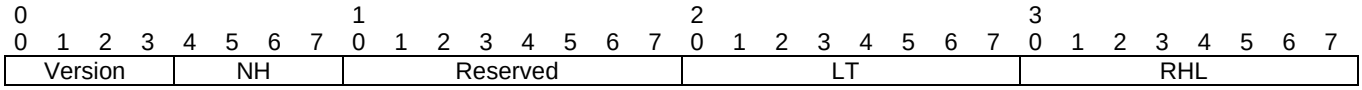

Figure 9: *Basic Header* format

8.6.2 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.
8	*LT*	Octet 2	Octet 2	8 bit unsigned integer	n/a	Lifetime field. Indicates the maximum tolerable time a packet can be buffered until it reaches its destination. Bit 0 to 5: LT sub-field Multiplier. Bit 6 to 7: LT sub-field Base. Encoded as specified in clause 8.6.4.
9	*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.

8.6.3 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 8.7
Secured Packet	2	GeoNetworking *Secured Packet* as specified in [i.5] "Intelligent Transport Systems (ITS); Security; Security header and certificate formats for ITS G5"

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

8.6.4 Encoding of the LT field

The *Lifetime (LT)* field shall indicate the maximum tolerable time a packet can 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 can 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`.

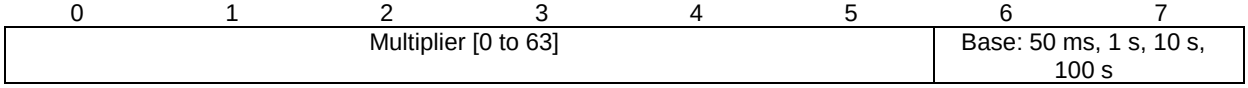

Figure 10: Composition of the *LT field*

8.7 Common Header

8.7.1 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.

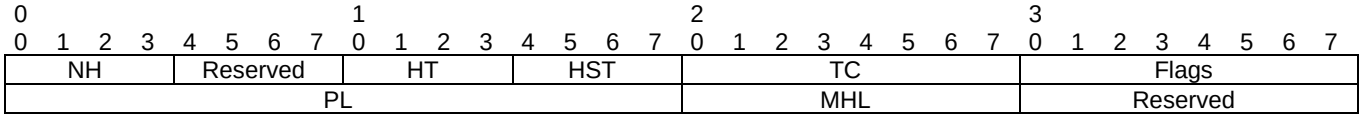

Figure 11: *Common Header* format

8.7.2 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 Facility-layer requirements on packet transport. Encoding is specified in clause 8.7.5.
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 7: Reserved. Set to 0.
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.						

8.7.3 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) as defined in EN 302 636-5-1 [5].
BTP-B	2	Transport protocol (BTP-B for non-interactive packet transport) as defined in EN 302 636-5-1 [5].
IPv6	3	IPv6 header as defined in EN 302 636-6-1 [6].

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

8.7.4 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

8.7.5 Encoding of the TC field

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

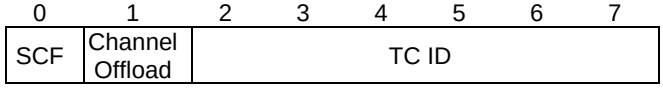

Figure 12: *Traffic Class (TC)* field composition

Table 10: Fields of TC field

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 neighbour exists (store-carry-forward). Length: 1 bit.
2	*Channel Offload*	Bit 1	Bit 1	Bit	n/a	Indicates whether the packet can be offloaded to another channel than specified in the *TC ID*. 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, e.g. in [i.2] for ITS-G5A. Length: 6 bits.

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

8.8 GeoNetworking packet header types

8.8.1 Overview

The following GeoNetworking packet header types are defined:

1) GUC packet header (clause 8.8.2).
2) TSB packet header (clause 8.8.3).
3) SHB packet header (clause 8.8.4).
4) GBC and GAC packet headers (clause 8.8.5).
5) BEACON packet header (clause 8.8.6).
6) LS Request and LS Reply packet headers (clauses 8.8.7 and 8.8.8).

8.8.2 GUC packet header

8.8.2.1 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*.

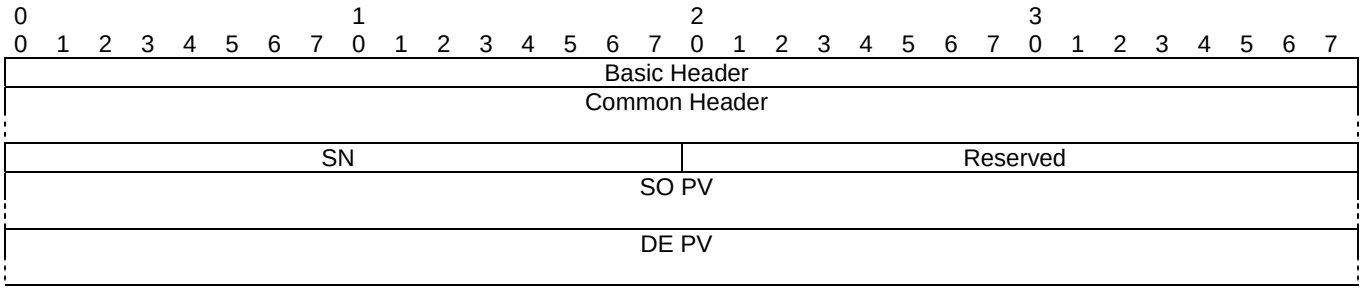

Figure 13: Packet header format: GUC

8.8.2.2 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 8.6 Length: 4 octets.
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 8.7. 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 7.3) 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 8.5.2 (Long Position Vector). Length: 24 octets.
6	*DE PV*	Octet 40	Octet 59	Short position vector	n/a	Short Position Vector containing the position of the destination. It shall consist of the fields as specified in clause 8.5.3 (Short Position Vector). Length: 20 octets.

8.8.3 TSB packet header

8.8.3.1 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*.

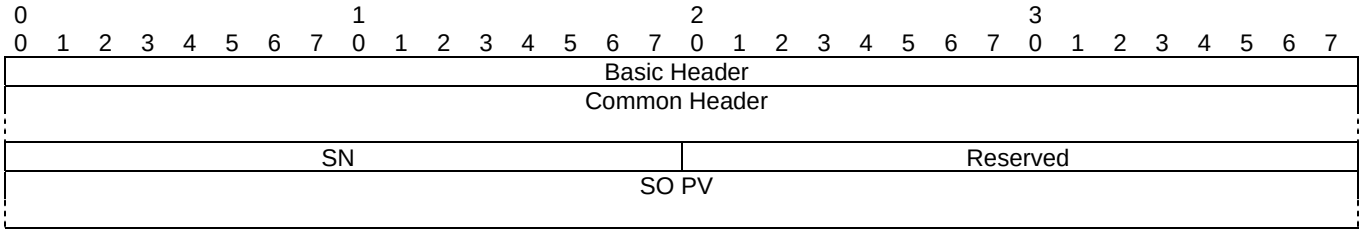

Figure 14: Packet header format: TSB

8.8.3.2 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 8.6 Length: 4 octets.
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 8.7. 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 7.3) 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 8.5.2 (Long Position Vector). Length: 24 octets.

8.8.4 SHB packet header

8.8.4.1 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*.

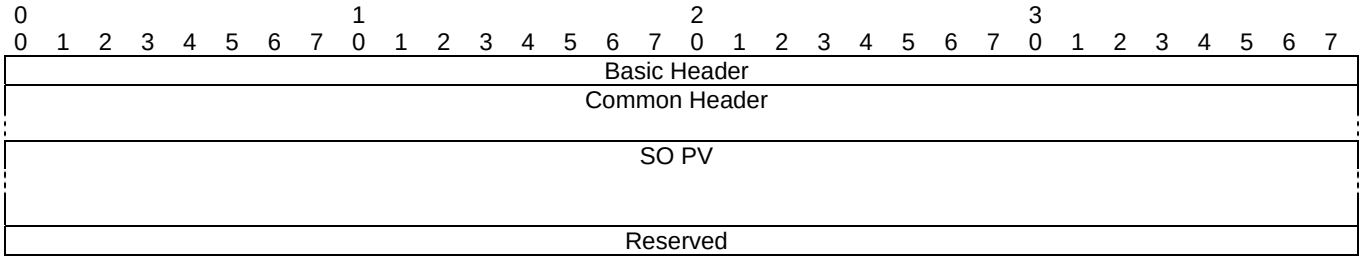

Figure 15: Packet header format: SHB

8.8.4.2 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 8.6 Length: 4 octets.
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 8.7. 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 8.5.2 (Long Position Vector). Length: 24 octets.
4	*Reserved*	Octet 36	Octet 39	32-bit unsigned integer	n/a	Reserved for media-dependent operations. If not used, it shall be set to 0. (see note)
NOTE: With ITS-G5 [i.2], the field is used to transmit DCC-related information.						

8.8.5 GBC/GAC packet header

8.8.5.1 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*.

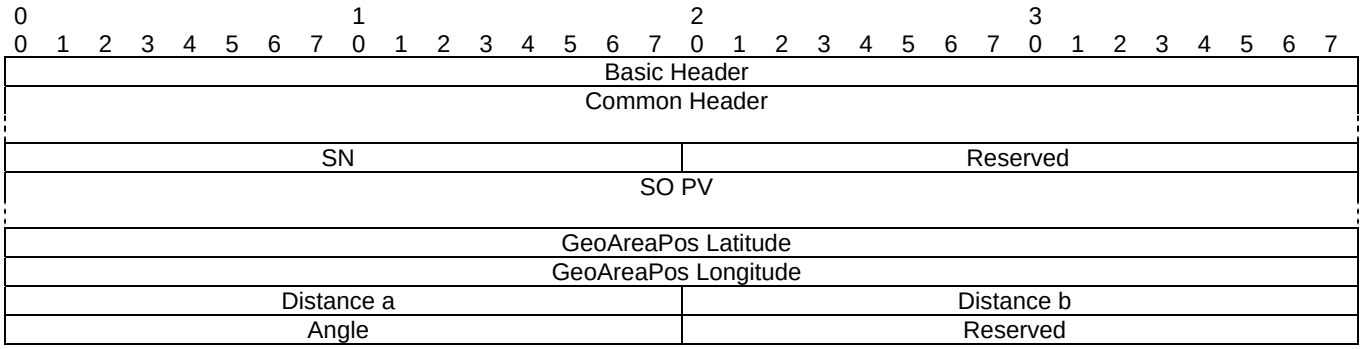

Figure 16: Packet header format: GBC/GAC

8.8.5.2 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 8.6 Length: 4 octets.
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 8.7. 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 7.3) 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 8.5.2 (Long Position Vector). Length: 24 octets.
6	*GeoAreaPos Latitude*	Octet 40	Octet 43	32-bit signed integer	[1/10 micro-degree]	WGS 84 [i.10] latitude for the centre position of the geometric shape as defined in [7] in 1/10 micro degree.
7	*GeoAreaPos Longitude*	Octet 44	Octet 47	32-bit signed integer	[1/10 micro-degree]	WGS 84 [i.10] longitude for the centre position of the geometric shape as defined in [7] 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 [7] in metres.
9	*Distance b*	Octet 50	Octet 51	16-bit unsigned integer	[m]	Distance b of the geometric shape as defined in [7] in metres.
10	*Angle*	Octet 52	Octet 53	16-bit unsigned integer	[°]	Angle of the geometric shape as defined in [7] 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.

8.8.6 BEACON packet header

8.8.6.1 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*.

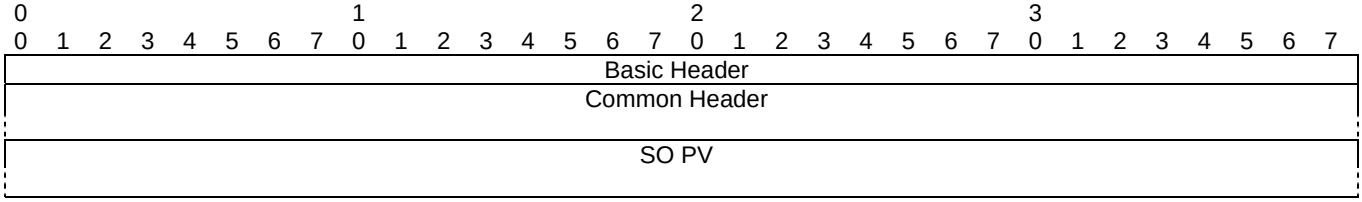

Figure 17: Packet header format: BEACON

8.8.6.2 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 8.6 Length: 4 octets.
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common header* as specified in clause 8.7. 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 8.5.2 (Long Position Vector). Length: 24 octets.

8.8.7 LS Request packet header

8.8.7.1 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*.

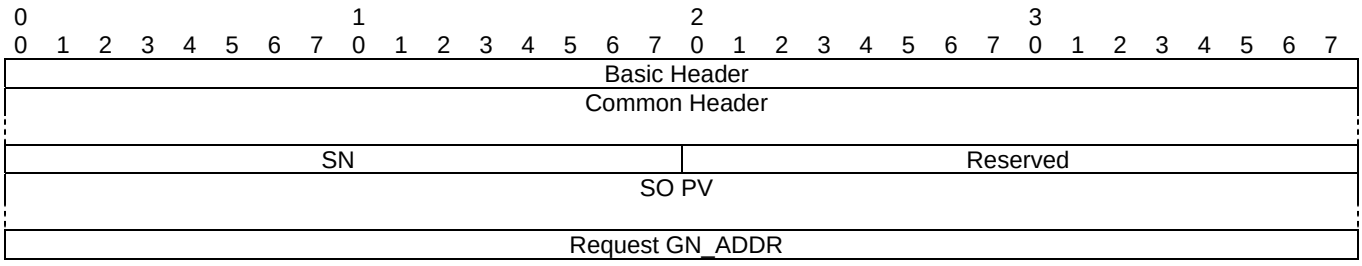

Figure 18: Packet header format: LS Request

8.8.7.2 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 8.6 Length: 4 octets.
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 8.7. 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 7.3) 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 8.5.2 (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.

8.8.8 LS Reply packet header

8.8.8.1 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*.

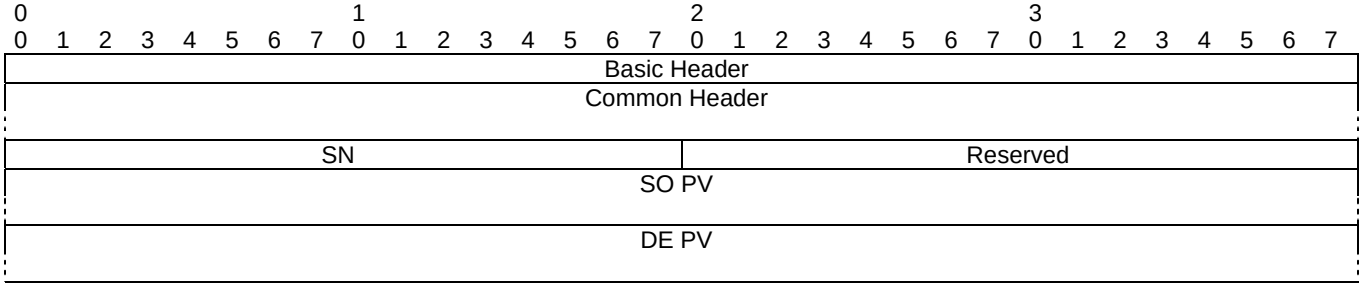

Figure 19: Packet header format: LS Reply

8.8.8.2 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 8.6 Length: 4 octets.
2	*Common Header*	Octet 4	Octet 11	Common Header	n/a	*Common Header* as specified in clause 8.7. 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 7.3) 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 8.5.2. (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 8.5.3 (Short Position Vector). Length: 20 octets.

9 Protocol operation

9.1 General

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

The operations include:

1) Network management:
   - Address configuration (clause 9.2.1),
   - Local position vector and time update (clause 9.2.2),
   - Beaconing (clause 9.2.3),
   - Location service (clause 9.2.4).
2) Packet handling:
   - GUC (clause 9.3.8),
   - TSB (clause 9.3.9),
   - SHB (clause 9.3.10),
   - GBC (clause 9.3.11),
   - GAC (clause 9.3.12).

9.2 Network management

9.2.1 Address configuration

9.2.1.1 General

At start-up, a GeoAdhoc router 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 9.2.1.2),

2) Managed address configuration (clause 9.2.1.3),

3) Anonymous address configuration (clause 9.2.1.4).

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 9.2.1.3.1) of the GeoAdhoc router can be updated (clause 9.2.1.3.2). In the anonymous address configuration, the address is configured by the security entity.

Operations for duplicate address detection are specified in clause 9.2.1.5.

9.2.1.2 Auto-address configuration

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

At start-up, the GeoAdhoc router 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).

9.2.1.3 Managed address configuration

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

With managed address configuration, the *ITS Networking & Transport Layer Management* layer is responsible for providing the MID field of the GeoAdhoc router address GN_ADDR.

9.2.1.3.1 Initial address configuration

At startup, the GeoAdhoc router shall request an MID field for the GN_ADDR from the *ITS Networking & Transport Layer Management* entity using the service primitive *GN-MGMT.request* (clause J.2). The *ITS Networking & Transport Layer Management* entity is responsible for generating the appropriate GeoNetworking address using the service primitive *GN-MGMT.response* (clause J.3).

9.2.1.3.2 Address update

The update of the MID field of the local GN_ADDR can be triggered by the GeoAdhoc router or the *ITS Networking & Transport Layer Management* entity.

If the update is triggered by the GeoAdhoc router, the GeoAdhoc router shall use the service primitive *GN-MGMT.request* (clause J.2). The *ITS Networking & Transport Layer Management* entity is responsible for generating the appropriate GeoNetworking address using the service primitive *GN-MGMT.response* (clause J.3).

If the update is triggered by the *ITS Networking & Transport Layer Management* entity, the *ITS Networking & Transport Layer Management* entity sends an unsolicited *GN-MGMT.response* to the GeoAdhoc router. Upon reception of the *GN-MGMT.response*, the GeoAdhoc router shall update its local GN_ADDR.

NOTE 1: For privacy reasons, the GN_ADDR can be derived from the current authorization ticket [i.3]). 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 can impair the performance of the GeoNetworking protocol.

9.2.1.4 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 Sec_GN_SAP interface and may be realized as SN-SAP [i.4].

In this method, the GeoNetworking protocol subscribes to the *ID-CHANGE-SUBSCRIBE* service at the security entity (clause K.4). In one possible implementation, it may register a callback function, which is executed when the pseudonym is changed.

At startup, the GeoAdhoc router 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* as specified in clause K.4;

2) process the service primitive *SN-IDCHANGE-SUBSCRIBE.confirm* that returns the subscription handle as specified in clause K.4;

3) process the service primitive *SN-IDCHANGE-EVENT.indication* that provides the parameter *id* as specified in clause K.5. The GeoAdhoc router shall set its local GN_ADDR to the parameter *id*;

4) generate the service primitive *SN-IDCHANGE-EVENT.response* as specified in clause K.5 to acknowledge the given command.

When the GeoAdhocRouter is shutdown or restarted, it 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* as specified in clause K.6;

2) process the service primitive *SN-IDCHANGE-UNSUBSCRIBE.confirm* as specified in clause K.6.

NOTE: Other services offered by the SN SAP, such as the SN-ID-LOCK, SN-ID-UNLOCK, SN-LOG-SECURITY-EVENT are not used in the present document.

9.2.1.5 Duplicate address detection

The configuration of the GeoNetworking address does not guarantee its uniqueness. In order to achieve uniqueness, a GeoAdhoc router shall execute the following operations for duplicate address detection:

1) Upon reception of a GeoNetworking packet, the GeoAdhoc router 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, corresponding to the 48-bit MAC address), with the sender link-layer address of the frame;

2) If a conflict is detected, the GeoNetworking protocol shall request a new MID field for the GeoNetworking address from the *ITS Networking & Transport Layer Management* entity using a service primitive *GN-MGMT.request* (clause J.2) indicating *Duplicate address* as the *Request cause*.

9.2.2 Local position vector and time update

9.2.2.1 Overview

Local position and time are set by the *ITS Networking & Transport Layer Management* entity via the GN_MGT interface (clause J.3).

9.2.2.2 Local Position Vector update

For position update, the *ITS Networking & Transport Layer Management* entity shall send an unsolicited *GN-MGMT.response* with the *Local position vector* parameter (clause J.3) to the GeoAdhoc router. Upon reception of the *GN-MGMT.response* with the *Local position vector* parameter, the GeoAdhoc router shall update its Local Position Vector (LPV) (clause 7.2).

As specified in clause 7.2.3, the LPV shall be updated with a minimum frequency of the GN protocol constant `itsGnMinUpdateFrequencyLPV`.

9.2.2.3 Time update

For time update, the *ITS Networking & Transport Layer Management* entity shall send an unsolicited *GN-MGMT.response* with the *Time* parameter (clause J.3) to the GeoAdhoc router. Upon reception of the *GN-MGMT.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: Details of the system time management and usage are implementation specific.

9.2.3 Beaconing

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

A BEACON packet shall be sent periodically unless the GeoAdhoc router sends another GeoNetworking packet that carries the GeoAdhoc router's LPV. At startup a GeoAdhoc router shall sent an initial beacon to announce its presence to other GeoAdhoc routers.

NOTE: 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.

9.2.4 Location service

The location service is used if a GeoAdhoc router needs to determine the position of another GeoAdhoc router. 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.

The location service function resides on top of the forwarding functions and can therefore use any forwarding type.

The location service is based on the exchange of control packets between GeoAdhoc routers (figure 20). 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.

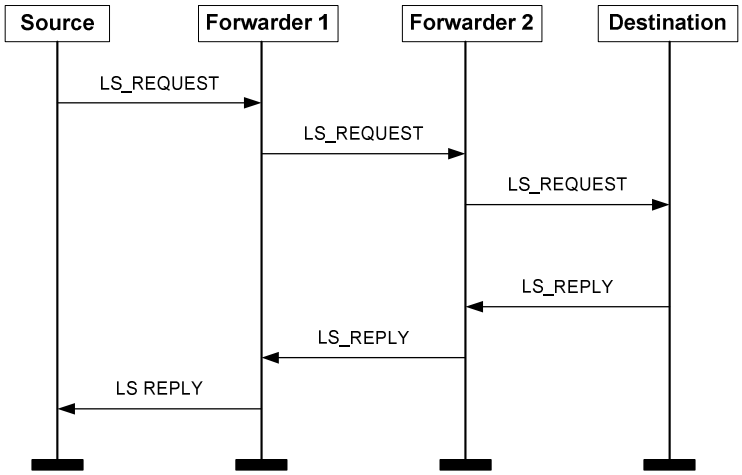

Figure 20: Message sequence chart for the location service (example scenario with two forwarders)

9.3 Packet handling

9.3.1 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 T/GN6-SDU, execute security functions, execute functions that are specific to the packet type, and pass the GN-PDU to the LL protocol entity via the IN interface.

The following packet handling types are defined:

1) Beacon packet handling (clause 9.3.6);
2) LS packet handling (clause 9.3.7);
3) GUC packet handling (clause 9.3.8);
4) TSB packet handling (clause 9.3.9);
5) SHB packet handling (clause 9.3.10);
6) GBC packet handling (clause 9.3.11); and
7) GAC packet handling (clause 9.3.12).

The packet handling is further specified in the following clauses.

9.3.2 *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 *GN-DATA.request* parameter *Security profile* indicates that the packet is unsecured. 2 (*Secured Packet*) if *GN-DATA.request* parameter *Security profile* indicates that the packet is secured.	Next header (table 5 in clause 8.6.3).
Reserved	Set to 0.	Reserved. Set to 0.
LT	Source: • Value of optional *Maximum packet lifetime* parameter from service primitive *GN-DATA.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 9.3.4). Source: • 1 if parameter *Packet transport type* in the service primitive *GN-DATA.request* is SHB, or if *Header type* HT = 1 (BEACON) • Value of optional *Maximum hop limit* parameter from service primitive *GN-DATA.request* • Otherwise GN protocol constant `itsGnDefaultHopLimit` if *GN-DATA.request* parameter *Packet transport type* is GUC, GBC, GBC or TSB Forwarder: Decrement RHL by one	Remaining hop limit.

9.3.3 Basic Header processing

When a GeoAdhoc router (forwarder, receiver and destination) processes a *Basic Header*, the GeoAdhoc router 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 is not equal to the GN protocol constant `itsGnProtocolVersion`, discard the packet and omit the execution of further steps;

NOTE 1: Future versions of the present document may relax this requirement and allow the processing packets with different version numbers that are backward compatible.

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 9.3.5;

   b) if NH = 2 (Secured Packet):

      i) execute the SN-DECAP service as specified in clause K.3 and 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*, *permissions* (clause K.3) carried in the *SN-DECAP.confirm* parameter (clause K.3) and proceed processing the *Common Header* as specified in clause 9.3.5;

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 *GN-DATA.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 ITS Facilities layer. Details are implementation specific.

9.3.4 *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 *GN-DATA.request* parameter *Upper protocol entity* is BTP-A 2 (BTP-B) if *GN-DATA.request* parameter *Upper protocol entity* is BTP-B 3 (IPv6) if *GN-DATA.request* parameter *Upper protocol entity* is IPv6	Next header (table 8 in clause 8.7.3)
Reserved	Set to 0.	Reserved.
HT	1 (BEACON) if the GeoNetworking packet is a Beacon 2 (GEOUNICAST) if *GN-DATA.request* parameter *Packet transport type* is GeoUnicast 3 (GEOANYCAST) if *GN-DATA.request* parameter *Packet transport type* is GeoAnycast 4 (GEOBROADCAST) if *GN-DATA.request* parameter *Packet transport type* is GBC 5 (TSB) if *GN-DATA.request* parameter *Packet transport type* is TSB 5 (TSB) if *GN-DATA.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 8.7.4)
HST	As specified in table 9 in clause 8.7.4	Header sub-type
TC	*Traffic class* defined in service primitive *GN-DATA.request* parameter *Traffic class* or GN protocol constant `itsGnDefaultTrafficClass` if the service primitive *GN-DATA.request* parameter *Traffic class* is not available.	Traffic class encoding specified in clause 8.7.5.
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 *GN-DATA.request* otherwise	Payload length in octets
MHL	• Source: • 1 if *GN-DATA.request* parameter *Packet transport type* is SHB or GeoNetworking packet is Beacon • Value of optional `Maximum hop limit` parameter from service primitive *GN-DATA.request* • GN protocol constant `itsGnDefaultHopLimit` if *GN-DATA.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 8.7.4).		

9.3.5 *Common Header* processing

When a GeoAdhoc router (forwarder, receiver, destination) processes a *Common Header*, the GeoAdhoc router 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 7.5) is not empty, forward the stored packets and remove them from the *BC forwarding packet buffer*;
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 9.3.6;
   c) if HT = 2 (GEOUNICAST) execute the steps specified in 9.3.8.3 and 9.3.8.4;
   d) if HT = 3 (GEOANYCAST) execute the steps specified in 9.3.12.3;
   e) if HT = 4 (GEOBROADCAST) execute the steps specified in 9.3.11.3;
   f) if HT = 5 (TSB) execute the steps specified in 9.3.9.3 (HST = MULTI_HOP) and 9.3.10.3 (HST = SINGLE_HOP);
   g) if HT = 6 (LS) execute the steps specified in 9.3.7.2 and 9.3.7.3.

9.3.6 Beacon packet handling

9.3.6.1 General

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

A BEACON packet shall be sent periodically unless the GeoAdhoc router sends another GeoNetworking packet that carries the GeoAdhoc router's LPV.

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

9.3.6.2 Source operations

At start-up, a GeoAdhoc router shall execute the following operations:

1) create a GN-PDU with a Beacon packet header (clause 8.8.6):
   a) set the fields of the *Basic Header* (clause 9.3.2);
   b) set the fields of the *Common Header* (clause 9.3.4);
   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 LPV as specified in clause 7.2	PV of the local 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* as specified in clause K.2 and the parameter setting in table 22;

Table 22: Parameter settings in the service primitive SN-ENCAP.request

Parameter name	Parameter setting
tbe_packet_length	Length of the Beacon.
tbe_packet	Common header + Beacon Extended header to be signed.
sec_profile	Not set.

   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 omit this operation;

   b) ITS-G5 then execute the operations as specified in [i.2];

4) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the Broadcast address of the LL entity;

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 8.8.6)

   a) set the fields of the *Basic Header* to the values specified in clause 9.3.2;

   b) set the fields of the *Common Header* to the values specified in clause 9.3.4;

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

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

   a) UNSPECIFIED then omit this operation;

   b) is set to ITS-G5 then execute the operations as specified in [i.2];

3) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the Broadcast address of the LL entity;

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

NOTE 2: The GeoAdhoc router resets the timer T_{Beacon} for every sent GeoNetworking packet that carries a LPV, i.e. a SHB packet.

9.3.6.3 Receiver operations

Receiver operations of Beacon packets are identical to the handling procedures of the SHB packet (clause 9.3.10.3) except step 8 (pass the payload of the GN-PDU to the upper protocol entity).

9.3.7 Location service packet handling

9.3.7.1 Source operations

9.3.7.1.1 Overview

Three cases are distinguished for the source operations:

1) source operation for initial LS Request (clause 9.3.7.1.2).

2) source operation for LS Request re-transmission (clause 9.3.7.1.3).

3) source operation for LS Reply (clause 9.3.7.1.4).

9.3.7.1.2 Source 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 7.4) and the execution of the next steps shall be omitted;

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

   a) send a service primitive *SN-ENCAP.request* as specified in clause K.2 and the parameter setting in table 23;

Table 23: Parameter settings in the service primitive SN-ENCAP.request

Parameter name	Parameter setting
tbe_packet_length	Length of the LS Request.
tbe_packet	Common header + LS Request Extended header to be signed.
sec_profile	Not set.

   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) issue a LS Request packet with a format as specified in clause 8.8.7 as a TSB packet:

   a) set the fields of the *Basic Header* to the values specified in clause 9.3.2;

   b) set the fields of the *Common Header* to the values specified in clause 9.3.4;

   c) set the fields of the LS Request *Extended Header* to the values specified in table 24;

Table 24: 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 7.3.	Sequence number of the packet.
SO PV	Actual values of the LPV as specified in clause 7.2.	Position vector containing the reference position of the local GeoAdhoc router (source of the GeoNetworking packet).
Request GN_ADDR	GN_ADDR from *GN-DATA.request* parameter *Destination (annex I).*	GeoNetworking address of the sought GeoAdhoc router.

4) start a timer T_{LS, GN_ADDR} with a timeout set to the value of the GN protocol constant `itsGnLocationServiceRetransmitTimer`;

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

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

9.3.7.1.3 Source operation for LS Request re-transmission

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

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 GeoAdhoc router shall:

      i) re- issue a LS Request packet with the format as specified in clause 8.8.7 as a TSB packet and the field setting for the LS Request *Extended Header* as specified in clause 9.3.7.1.2, table24;

      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. $RTC_{LS,\ GN\_ADDR} \geq$ `itsGnLocationServiceMaxRetrans`, the GeoAdhoc router shall:

      i) remove the pending packets for the sought GN_ADDR from the *LS packet buffer* (clause 7.4);

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

9.3.7.1.4 Source 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 9.3.3);

2) *Common Header* processing (clause 9.3.5);

3) execute duplicate packet detection 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 7.4) 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 7.4);

      ii) set *SO LS_pending* to false;

   b) if the *UC forwarding packet buffer* (clause 7.5) 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 $RTC_{LS,\ GN\_ADDR}$.

9.3.7.2 Forwarder operations

If a GeoAdhoc router receives a LS Request packet and the *Request GN_ADDR* field in the LS Request header does not match its *GN_ADDR*, the GeoAdhoc router shall handle the packet according to the packet handling procedure for TSB (clause 9.3.9.3), except step 7 for passing the payload of the GN-PDU to the upper protocol entity.

If a GeoAdhoc router 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 GeoAdhoc router shall handle the packet according to the packet handling operations (forwarder) for GUC (clause 9.3.8.3).

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

9.3.7.3 Destination operations

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

1) *Basic Header* processing (clause 9.3.3);

2) *Common Header* processing (clause 9.3.5);

3) execute duplicate packet detection 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 duplicate address detection as specified in clause 9.2.1.5;

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 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 GAC *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) issue a LS Reply packet as a GUC packet (clause 8.8.2) and forward the packet according to the forwarding procedure for GUC (clause 9.3.8.3).

9.3.8 GUC packet handling

9.3.8.1 General

This clause specifies the operations of a GeoAdhoc router 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 `itsGnGeoUnicastForwardingAlgorithm` and specified in annex D.

9.3.8.2 Source operations

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

1) 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 9.2.4 and omit the execution of further steps. Otherwise, the source shall proceed with step 2;

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 *GN-DATA.request* parameter *Traffic class* is enabled:

   a) buffer the GUC packet in the *UC forwarding packet buffer* and omit the execution of further steps;

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

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

3) determine the link-layer address *LL_ADDR_NH* of the next hop (annex D):

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

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

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

4) 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;

NOTE 3: The CBF algorithm (clause D.3) returns the Broadcast LL or, in case of no neighbours, 0 (see clause D.3).

5) create a GN-PDU with the T/GN6-SDU as payload and a GUC packet header (clause 8.8.2):

   a) set the fields of the *Basic Header* (clause 9.3.2);

   b) set the fields of the *Common Header* (clause 9.3.4);

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

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

Field name	Field setting	Description
SN	Actual value of the local sequence number (clause 7.3).	Sequence number of the packet
SO PV	Actual values of the LPV (clause 7.2).	Position vector containing the reference position of the local 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

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

   a) send a service primitive *SN-ENCAP.request* as specified in clause K.2 and the parameter setting in table 26;

Table 26: Parameter settings in the service primitive SN-ENCAP.request

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 *GN-DATA.request [i.2]*.

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 GN-DATA.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 4: The maximum repetition time of the packet is specified in the *Maximum repetition time* parameter of the service primitive *GN-DATA.request*.

NOTE 5: For every retransmission, the source operations need to be re-executed.

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

a) UNSPECIFIED then omit this operation;

b) ITS-G5 then execute the operations as specified in [i.2];

9) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the LL address of the next hop *LL_ADDR_NH*.

9.3.8.3 Forwarder 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 does not match its *GN_ADDR*, the GeoAdhoc router shall execute the following operations:

1) *Basic Header* processing (clause 9.3.3);

2) *Common Header* processing (clause 9.3.5);

3) if the GN protocol constant `itsGnGeoUnicastForwardingAlgorithm` is set to 0 (UNSPECIFIED) or to 1 (GREEDY), execute duplicate packet detection 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 (`itsGnGeoUnicastForwardingAlgorithm` is set to 2), the algorithm relies on the processing of duplicate packets and their handling is part of the forwarding algorithm.

4) execute duplicate address detection as specified in clause 9.2.1.5;

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 exists or the IS_NEIGHBOR 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 DE PV fields with the PV(DE) in the LocT (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, 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.

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

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

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 7.4);

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

   b) if the *UC forwarding packet buffer* (clause 7.5) 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:

   a) if the packet is unsecured, i.e. NH = 1 (Common Header), buffer the GUC packet in the *UC forwarding packet buffer* and omit the execution of further steps;

   b) if the packet is secured, i.e. NH = 2 (Secured Packet), buffer the secured GUC packet in the *UC forwarding packet buffer* and omit the execution of further steps;

NOTE 7: If *SCF* for the traffic class is disabled, the GUC packet is never buffered but sent immediately with a broadcast MAC destination address.

NOTE 8: Buffered packets may be further processed when the GeoAdhoc router receives a packet (e.g. SHB, GBC, BEACON, etc.), see the corresponding clauses on forwarder and receiver operations.

12) determine the link-layer address *LL_ADDR_NH* of the next hop (annex D):

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

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

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

13) 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;

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

   a) UNSPECIFIED, omit this operation;

b) ITS-G5, execute the operations as specified in [i.2];

15) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the LL address of the next hop LL_ADDR_NH.

9.3.8.4 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 9.3.3);

2) *Common Header* processing (clause 9.3.5);

3) execute duplicate packet detection 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 duplicate address detection as specified in clause 9.2.1.5;

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 7.4);

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

   b) if the *UC forwarding packet buffer* (clause 7.5) 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 *GN-DATA.indication* with the parameter settings in table 27.

Table 27: Parameter settings in the service primitive *GN-DATA.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 8.7.3)
Packet transport type	GeoUnicast
Destination	*DE GN_ADDR*
Source position vector	Values of *SO PV* from *Extended Header*
Security report	Result of the decrypt and verify operation in the service primitive *SN-DECAP.indication* parameter *report* (clause K.3)
Certificate id	Identification of the source certificate, for example the certificate hash, in the service primitive *SN-DECAP.indication* parameter *certificate_id* (clause K.3) (optional)
Permissions	Permissions of the source certificate in the service primitive *SN-DECAP.indication* parameter *permissions* if reported by the SN-DECAP service (clause K.3) (optional)
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

9.3.9 TSB packet handling

9.3.9.1 General

This clause specifies the operations of a GeoAdhoc router 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.

9.3.9.2 Source operations

On reception of a service primitive *GN-DATA.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 8.8.3):

   a) set the fields of the *Basic Header* (clause 9.3.2);

   b) set the fields of the *Common Header* (clause 9.3.4);

   c) set the fields of the TSB *Extended Header* (table 28);

Table 28: Field settings for the TSB *Extended Header*

Field name	Field setting	Description
SN	Actual value of the local sequence number (clause 7.3)	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 LPV (clause 7.2)	Position vector containing the reference position of the local GeoAdhoc router (source of the GeoNetworking packet)

2) if the optional Security profile parameter in the service primitive GN-DATA.request is set:

   a) send a service primitive *SN-ENCAP.request* as specified in clause K.2 and the parameter setting in table 29;

Table 29: Parameter settings in the service primitive SN-ENCAP.request

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 parameter *Security profile* in the service primitive *GN-DATA.request [*i.2*]*.

   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 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 *GN-DATA.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 MAC destination address.

NOTE 2: Buffered packets may be further processed when the GeoAdhoc router 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 GN-DATA.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 *GN-DATA.request*.

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

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

   a) UNSPECIFIED then omit this operation;

   b) is set to ITS-G5 then execute the operations as specified in [i.2];

6) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the Broadcast address of the LL entity.

9.3.9.3 Forwarder and receiver operations

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

1) *Basic Header* processing (clause 9.3.3);

2) *Common Header* processing (clause 9.3.5);

3) execute duplicate packet detection 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 duplicate address detection as specified in clause 9.2.1.5;

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 *GN-DATA.indication* with the parameter settings in table 30;

Table 30: Parameter settings in the service primitive *GN-DATA.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 8.7.3)
Packet transport type	TSB
Source position vector	Values of *SO PV* from *Extended Header*
Security report	Result of the decrypt and verify operation in the service primitive *SN-DECAP.indication* parameter *Report* (clause K.3)
Certificate id	Identification of the source certificate, for example the certificate hash, in the service primitive *SN-DECAP.indication* parameter *certificate_id* (clause K.3) (optional)
Permissions	Permissions of the source certificate in the service primitive *SN-DECAP.indication* parameter *permissions* if reported by the SN-DECAP service (clause K.3) (optional)
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 7.4);

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

b) if the *UC forwarding packet buffer* (clause 7.5) 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:

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 MAC destination address.

NOTE 3: Buffered packets may be further processed when the GeoAdhoc router 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 omit this operation;

   b) ITS-G5 then execute the operations as specified in [i.2];

12) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the Broadcast address of the LL entity.

9.3.10 SHB packet handling

9.3.10.1 General

This clause specifies the operations of a GeoAdhoc router 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.

9.3.10.2 Source operations

On reception of a service primitive *GN-DATA.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 8.8.4):

   a) set the fields of the *Basic Header* (clause 9.3.2);

   b) set the fields of the *Common Header* (clause 9.3.4);

   c) set the fields of the SHB *Extended Header* (table 31);

Table 31: Field settings for the SHB *Extended Header*

Field name	Field setting	Description
SO PV	Actual values of the LPV as specified in clause 7.2	PV of the local 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 GN-DATA.request is set:

   a) send a service primitive *SN-ENCAP.request* as specified in clause K.2 and the parameter setting in table 32;

Table 32: Parameter settings in the service primitive SN-ENCAP.request

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 parameter *Security profile* in the service primitive *GN-DATA.request [i.2].*

   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 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 *GN-DATA.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 MAC destination address.

NOTE 2: Buffered packets may be further processed when the GeoAdhoc router 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 GN-DATA.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 *GN-DATA.request*.

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

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

   a) UNSPECIFIED then omit this operation;

   b) ITS-G5 then execute the operations as specified in [i.2];

6) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the Broadcast address of the LL entity;

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

9.3.10.3 Receiver operations

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

1) *Basic Header* processing (clause 9.3.3);

2) *Common Header* processing (clause 9.3.5);

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

4) execute duplicate address detection as specified in clause 9.2.1.5;

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

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

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

8) pass the payload of the GN-PDU to the upper protocol entity by means of a service primitive *GN-DATA.indication* with the parameter settings in table 33;

Table 33: Parameter settings in the service primitive *GN-DATA.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 8.7.3)
Packet transport type,	SHB
Source position vector	Values of *SO PV* from SHB *Common Header*
Security report	Result of the decrypt and verify operation in the service primitive *SN-DECAP.indication* parameter *Report* (clause K.3)
Certificate id	Identification of the source certificate, for example the certificate hash, in the service primitive *SN-DECAP.indication* parameter *certificate_id* (clause K.3) (optional)
Permissions	Permissions of the source certificate in the service primitive *SN-DECAP.indication* parameter *permissions* if reported by the SN-DECAP service (clause K.3) (optional)
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

9) 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 7.5) for the *SO* is not empty, forward the stored packets and remove them from the *UC forwarding packet buffer*.

9.3.11 GBC packet handling

9.3.11.1 General

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

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

9.3.11.2 Source operations

On reception of a service primitive *GN-DATA.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 8.8.5):

   a) set the fields of the *Basic Header* (clause 9.3.2);

   b) set the fields of the *Common Header* (clause 9.3.4);

   c) set the fields of the GBC *Extended Header* (table 34).

Table 34: Field settings for the GBC *Extended Header*

Field name	Field setting	Description
SN	Actual value of the local sequence number (clause 7.3)	Sequence number of the packet
Reserved	Set to 0.	Reserved.
SO PV	Actual values of the LPV (clause 7.2)	Position vector containing the reference position of the local GeoAdhoc router (source of the GeoNetworking packet)
GeoAreaPos Latitude	GeoAreaPos Latitude from service primitive *GN-DATA.request*	GeoArea Area specification according to EN 302 931 [7].
GeoAreaPos Longitude	GeoAreaPos Longitude from service primitive *GN-DATA.request*	
Distance a	Distance a from service primitive *GN-DATA.request*	
Distance b	Distance a from service primitive *GN-DATA.request*	
Angle	Angle from service primitive *GN-DATA.request*	
Reserved	Set to 0.	Reserved.

2) if the optional Security profile parameter in the service primitive GN-DATA.request is set:

   a) send a service primitive *SN-ENCAP.request* as specified in clause K.2 and the parameter setting in table 35;

Table 35: Parameter settings in the service primitive SN-ENCAP.request

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_profile	The value of the parameter *Security profile* in the service primitive *GN-DATA.request [i.2]*.

   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 neighbour exists, i.e. the LocT does not contain a LocTE with the *IS_NEIGHBOUR* flag set to TRUE, and *SCF* flag for the traffic class in the service primitive *GN-DATA.request* parameter *Traffic class* is set, then buffer the GBC 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 GBC packet is never buffered but sent immediately with a broadcast MAC destination address.

NOTE 2: Buffered packets may be further processed when the GeoAdhoc router 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 GN-DATA.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 3: The maximum repetition time of the packet is specified in the *Maximum repetition time* parameter of the service primitive *GN-DATA.request*.

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

5) determine the link-layer address *LL_ADDR_NH* of the next hop (annex E):

   a) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 0 (UNSPECIFIED), execute the Simple GeoBroadcast with line forwarding algorithm as specified in clause E.2;

   b) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 1 (SIMPLE), execute the Simple GeoBroadcast with line forwarding algorithm as specified in clause E.2;

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

   d) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 3 (ADVANCED), execute the Advanced forwarding algorithm as specified in clause E.4;

6) 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;

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

   a) UNSPECIFIED then omit this operation;

   b) ITS-G5 then execute the operations as specified in [i.2];

8) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the LL address of the next hop *LL_ADDR_NH*.

9.3.11.3 Forwarder and receiver operations

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

1) *Basic Header* processing (clause 9.3.3);

2) *Common Header* processing (clause 9.3.5);

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

NOTE 1: For CBF and the Advanced forwarding algorithm (`itsGnGeoBroadcastForwardingAlgorithm` 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 duplicate address detection as specified in clause 9.2.1.5;

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 [7] 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 *GN-DATA.indication* with the parameter settings in table 36;

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

Table 36: Parameter settings in the service primitive *GN-DATA.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 8.7.3)
Packet transport type,	GeoBroadcast
Destination	GeoArea (GeoPos, distance a, distance b, angle)
Source position vector	Values of *SO PV* from *Extended Header*
Security report	Result of the decrypt and verify operation in the service primitive *SN-DECAP.indication* parameter *Report* (clause K.3)
Certificate id	Identification of the source certificate, for example the certificate hash, in the service primitive *SN-DECAP.indication* parameter *certificate_id* (clause K.3) (optional)
Permissions	Permissions of the source certificate in the service primitive *SN-DECAP.indication* parameter *permissions* if reported by the SN-DECAP service (clause K.3) (optional)
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 7.4);

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

   b) if the *UC forwarding packet buffer* (clause 7.5) 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 enabled:

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

NOTE 4: If *SCF* for the traffic class is disabled, the GBC packet is never buffered but sent immediately with a broadcast MAC destination address.

11) determine the link-layer address LL_ADDR_NH of the next hop (annex E):

    a) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 0 (UNSPECIFIED), execute the Simple GeoBroadcast with line forwarding algorithm as specified in clause E.2;

    b) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 1 (SIMPLE), execute the Simple GeoBroadcast with line forwarding algorithm as specified in clause E.2;

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

    d) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 3 (ADVANCED), execute the Advanced forwarding algorithm as specified in clause E.4;

12) if *LL_ADDR_NH* = 0 and *SCF* for the traffic class in the *TC* field of the *Common Header* is enabled, then buffer the GBC packet in the *BC forwarding packet buffer* and omit the execution of further steps;

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

NOTE 6: Buffered packets may be further processed when the GeoAdhoc router 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 omit this operation;

    b) ITS-G5 then execute the operations as specified in [i.2];

14) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the LL address of the next hop LL_ADDR_NH.

9.3.12 GAC packet handling

9.3.12.1 General

This clause specifies the operations of a GeoAdhoc router 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.

9.3.12.2 Source operations

The operations of the source of a GBC packet are identical with the source of a GeoBroadcast packet as specified in clause 9.3.11.2, except the operation in step 5) (7) determine function F(x,y) as specified in [7] clause 5). Instead, the source shall execute the following operation:

5) determine function F(x,y) as specified in [7] clause 5:

    a) if $F(x, y) < 0$ (GeoAdhoc router is outside the geographical area) and the GN protocol constant `itsGnGeoAreaLineForwarding` is set to TRUE, execute the GeoUnicast forwarding algorithm and determine the link-layer address *LL_ADDR_NH* of the next hop (annex D);

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

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

        iii) if the GN protocol constant `itsGnGeoUnicastForwardingAlgorithm` is set to 2 (CBF), execute the CBF algorithm as specified in clause D.3;

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

i) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 0 (UNSPECIFIED), execute the Simple GeoBroadcast with line forwarding algorithm as specified in clause E.2;

ii) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 1 (SIMPLE), execute the Simple GeoBroadcast with line forwarding algorithm as specified in clause E.2;

iii) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 2 (CBF), execute the Contention-based forwarding algorithm as specified in clause E.3;

iv) if the GN protocol constant `itsGnGeoBroadcastForwardingAlgorithm` is set to 3 (ADVANCED), execute the Advanced forwarding algorithm as specified in clause E.4.

NOTE: The procedure in case b) (GeoAdhoc router is inside or at the border of the geographical area) ensures that the GAC packet is sent at least once.

9.3.12.3 Forwarder and receiver operations

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

1) *Basic Header* processing (clause 9.3.3);

2) *Common Header* processing (clause 9.3.5);

3) execute duplicate packet detection 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 duplicate address detection as specified in clause 9.2.1.5;

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 [7] 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 7.4);

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

b) if the *UC forwarding packet buffer* (clause 7.5) 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 *GN-DATA.indication* with the parameter settings in table 37;

Table 37: Parameter settings in the service primitive *GN-DATA.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 8.7.3)
Packet transport type,	GeoAnycast
Destination	GeoArea (GeoPos, distance a, distance b, angle)
Source position vector	Values of SO PV from *Extended Header*
Security report	Result of the decrypt and verify operation in the service primitive *SN-DECAP.indication* parameter *Report* (clause K.3)
Certificate id	Identification of the source certificate, for example the certificate hash, in the service primitive *SN-DECAP.indication* parameter *certificate_id* (clause K.3) (optional)
Permissions	Permissions of the source certificate in the service primitive *SN-DECAP.indication* parameter *permissions* if reported by the SN-DECAP service (clause K.3) (optional)
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 the GN protocol constant `itsGnGeoAreaLineForwarding` is set to TRUE, execute the GeoUnicast forwarding algorithm and determine the link-layer address *LL_ADDR_NH* of the next hop (annex D):

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

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

iii) if the GN protocol constant `itsGnGeoUnicastForwardingAlgorithm` is set to 2 (CBF), execute the CBF algorithm as specified in clause D.3.

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 LL_ADDR_NH = 0 and SCF for the traffic class in the TC field of the Common Header is enabled, then buffer the GBC packet in the BC forwarding packet buffer and omit the execution of further steps;

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

a) UNSPECIFIED then omit this operation;

b) ITS-G5 then execute the operations as specified in [i.2];

13) pass the GN-PDU to the LL protocol entity via the IN interface and set the destination address to the LL address of the next hop LL_ADDR_NH.

10 Conformance and test methods

Conformance and test methods for GeoNetworking are not specified in the present document.

Annex A (normative): Duplicate packet detection

A.1 General

A GeoAdhoc router can receive multiple copies of the same packet. Reasons for packet duplications can 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.

The present document specifies the following methods for duplicate packet detection:

1) Sequence number and timestamp-based (clause A.2).

2) Timestamp-based (clause A.3).

The GeoNetworking protocol applies the sequence number and timestamp-based method for duplicate packet detection (clause A.2) to multi-hop packets (GUC, TSB, GBC, GAC, LS Request, LS Reply). The timestamp-based method (clause A.3) is applied to GeoNetworking packets that do not carry a SN field, i.e. single-hop packets (BEACON and SHB).

A.2 SN- and TST-based duplicate packet detection

For the SN- and TST-based method for duplicate packet detection, a GeoAdhoc router maintains the sequence number and timestamp of the last packet from the source that was identified as 'not duplicated' in its LocT, i.e. SN(GN_ADDR) and TST(GN_ADDR) as defined in clause 7.1.2 where GN_ADDR=SO is the GeoNetworking address of the source. When the GeoAdhoc router processes a GeoNetworking packet, it compares the value of the SN field carried in the GeoNetworking packet SN(P) and SN(SO), and also the value of the TST. If SN(P) is greater than SN(SO), then the received packet is regarded as 'not duplicated' and SN(SO) is updated.

NOTE: The GeoNetworking protocol does not provide packet re-ordering. Due to the simple duplicate packet detection, out-of-sequence packets will be discarded.

The sequence numbers and timestamps in the GeoNetworking protocol are limited in the number of bits. In order to handle the wraparound of sequence numbers (that the sequence number is incremented from the maximum possible value to zero) and timestamp, respectively, the following algorithm shall be used:

```
1     -- P is the received GeoNetworking packet
2     -- SN(P) is the sequence number in the received GeoNetworking packet
3     -- SN(SO) is the last received sequence number from source SO
4     -- SN_MAX is the maximum sequence number = 2^16 - 1
5     -- TST(P) is the timestamp in the received GeoNetworking packet
6     -- TST(SO) is the last received timestamp from source SO
7     -- TST_MAX is the maximum value of the timestamp = 2^32 - 1
8
9     IF (((TST(P) > TST(SO) AND ((TST(P) – TST(SO)) <= TST_MAX/2)) OR
              ((TST(SO) > TST(P)) AND ((TST(SO) - TST(P)) > TST_MAX/2))) THEN
10                                                  # TST(P) is greater than TST(SO)
11        TST(SO) ← TST(P)
12        SN(SO) ← SN(P)                            # P is not a duplicate packet
13    ELSEIF TST(P) = TST(SO) THEN
14        IF (((SN(P) > SN(SO) AND ((SN(P) – SN(SO)) <= SN_MAX/2)) OR
               ((SN(SO) > SN(P)) AND ((SN(SO) - SN(P)) > SN_MAX/2))) THEN
15                                                  # SN(P) is greater than SN(SO)
16            TST(SO) ← TST(P)
17            SN(SO) ← SN(P)                        # P is not a duplicate packet
18        ELSE
19                                                  # SN(P) is not greater than SN(SO)
20                                                  # P is a duplicate
21        ENDIF
22    ELSE
23                                                  # TST(P) not greater than TST(SO)
24    ENDIF
```

A.3 TST-based duplicate packet detection

The TST-based method for duplicate packet detection uses the timestamp carried in the SO PV of the *Extended Header* to identify a duplicate packet. The following algorithm shall be used:

```
1     -- P is the received GeoNetworking packet
2     -- TST(P) is the timestamp in the received GeoNetworking packet
3     -- TST(SO) is the last received timestamp from source SO
4     -- TS_MAX is the maximum value of the timestamp = 2^32 - 1
5
6     IF (((TST(P) > TST(SO) AND ((TST(P) – TST(SO)) <= TST_MAX/2)) OR
              ((TST(SO) > TST(P)) AND ((TST(SO) - TST(P)) > TST_MAX/2))) THEN
7                                           # TST(P) is greater than TST(SO)
8         TST(SO) ← TST(P)                  # P is not a duplicate packet
9     ELSE
10                                          # P is a duplicate
11    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 9.3) to control that a GeoAdhoc router does not exceed a predefined packet rate or geographical area size.

B.2 Packet data rate control

A GeoAdhoc router 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:

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

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.

β Weight factor ($0 < ß < 1$), set to 0,5.

If the packet data rate of a GeoAdhoc router 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 9.3) 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 7.1.3).

```
1     -- RP is the received GeoNetworking packet
2     -- PV_RP is the position vector in the received GeoNetworking packet
3     -- PV_LocT is the position vector in the LocT to be updated
4     -- TST_PV,RP is the timestamp for the position vector in the received
5     --       GeoNetworking packet
6     -- TST_PV,LocT is the timestamp for the position vector in the location table
7     --       to be updated
8     -- TS_Max is the maximum value of the timestamp = 2^32-1
9     -- T(LocTE)is the lifetime of the location table entry
10    -- itsGnLifetimeLocTE is the value of the GN protocol constant itsGnLifetimeLocTE
11    IF (((TST_PV,RP > TST_PV,LocT) AND ((TST_PV,RP - TST_PV,LocT) <= TST_Max/2)) OR
12        ((TST_PV,LocT > TST_PV,RP) AND ((TST_PV,LocT - TST_PV,RP) > TST_Max/2))) THEN
13        TST_PV,RP is greater than TST_PV,LocT
14        PV_LocT ← PV_RP
15        T(LocTE) ← value(itsGnLifetimeLocTE)
16    ELSE
17            TST_PV,RP is not greater than TST_PV,LocT
18    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:

```
1     -- FP is the GeoNetworking packet to be forwarded
2     -- PV_FP is the position vector in the GeoNetworking packet to be forwarded
3     -- PV_LocT is the position vector in the LocT
4     -- TST_PV,FP is the timestamp for the position vector in the GeoNetworking
5     --       packet to be forwarded
6     -- TST_PV,LocT is the timestamp for the position vector in the location table
7     -- TS_Max is the maximum value of the timestamp = 2^32-1
8     IF (((TST_PV,LocT > TST_PV,FP) AND ((TST_PV,LocT - TST_PV,FP) <= TST_Max/2)) OR
9     ((TST_PV,FP > TST_PV,LocT) AND ((TST_PV,FP - TST_PV,LocT) > TST_Max/2))) THEN
10        TST_PV,LocT is greater than TST_PV,FP
11            PV_FP ← PV_LocT
12    ELSE
13            TST_PV,FP is not greater than TST_PV,LocT
14    ENDIF
```

NOTE: The algorithm is used in to update the DE PV fields with the PV(DE) in the LocT (clause 9.3.8.3 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): GeoUnicast forwarding algorithms

D.1 Overview

The GeoUnicast forwarding algorithm is executed by a GeoAdhoc router to relay a packet to the next hop.

The present document defines two GeoUnicast forwarding algorithms:

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

2) Contention-based forwarding (CBF) algorithm (clause D.3).

D.2 Greedy Forwarding algorithm for GeoUnicast

With the Greedy Forwarding (GF) algorithm, the GeoAdhoc router uses the location information of the destination carried in the GUC 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 GUC packet is forwarded.

The algorithm returns one of the following two values:

- the LL address of the next hop NH_LL_ADDR,
- 0 indicates that no forwarder could be found and the packet is buffered in the *UC forwarding packet buffer*.

The pseudo-code of the algorithm is below:

```
1     -- P is the GUC packet to be forwarded
2     -- i is the i-th LocTE
3     -- NH is the LocTE idenfified as next hop, NH.LL_ADDR its link layer address
4     -- NH_LL_ADDR is the link layer address of the next hop
5     -- LPV is the local position vector
6     -- PV_P is the destination position vector in the GeoNetworking packet to be forwarded
7     -- PV_I is the position vector of the i-th LocTE
8     -- MFR indicates the progress according the to MFR policy
9     -- B is the UC forwarding packet buffer
10
11    MFR = DIST(PV_P, LPV)                              Initialize MFR
12    FOR (i∈ LocT)
13        IF (i.IS_NEIGHBOUR) THEN                       # LocTE i is neighbour
14            IF (DIST(PV_P, PV_I) < MFR) THEN
15                NH ← i
16                MFR ← DIST(PV_P, PV_I)
17            ENDIF
18        ENDIF
19    ENDFOR
20    IF (MFR < DIST(PV_P, LPV)) THEN
21        SET NH_LL_ADDR ← NH.LL_ADDR
22    ELSE                                               # Forwarder is at a local optimum
23        ADD P TO B
24        SET NH_LL_ADDR ← 0                             # Indicates that packet is buffered
25    ENDIF
26    RETURN NH_LL_ADDR
```

NOTE: If no neighbour with greater progress than the local GeoAdhoc router exists, the packet has reached a local optimum and the result '0' is returned indicating that no forwarder could be found.

D.3 Contention-Based Forwarding algorithm for GeoUnicast

With the Contention-Based Forwarding (CBF) algorithm, a receiver decides to be a forwarder of a GUC packet. This is contrary to the sender-based forwarding scheme specified in clause D.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 GUC 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 D.1).

Equation D.1 Calculation of timeout TO_CBF_GUC for buffering packets in the CBF packet buffer:

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

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 `itsGnGeoUnicastCbfMinTime` and `itsGnGeoUnicastCbfMaxTime`, respectively. If DIST_MAX is not defined in the specification of GeoNetworking media-dependent functionality for the specific ITS access technology (e.g. TS 102 636-4-2 [i.2]), it shall be set to the GN protocol constant `itsGnDefaultMaxCommunicationRange`.

Upon expiration of the timer, the GeoAdhoc router re-broadcasts the GUC 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 GUC packet from the CBF packet buffer.

NOTE 2: Compared to the GF algorithm (clause D.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 LL 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 D.1 for illustration.

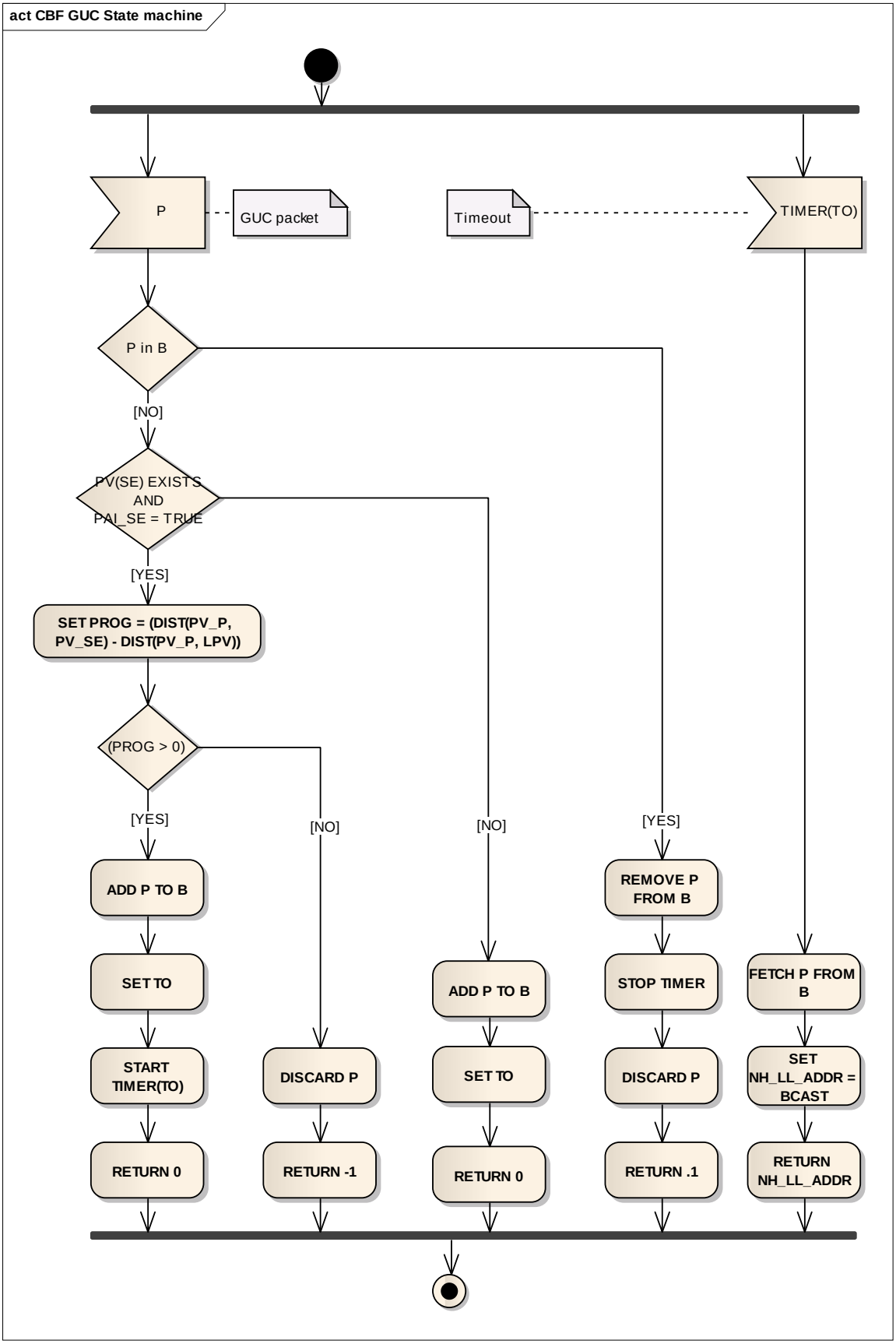

Figure D.1: GeoUnicast CBF activity diagram

The pseudo-code of the algorithm is below:

```
1       -- P is the GUC packet to be forwarded
2       -- LPV is the local position vector
3       -- PV_P is the destination position vector contained in the GeoNetworking packet
4       -- PV_SE is the sender position vector in the LocT with position accuracy indicator PAI_SE
5       -- B is the CBF packet buffer
6       -- TO is the timeout that triggers the re-broadcast of the packet
7       -- NH_LL_ADDR is the LL address of the next hop
8       -- BCAST is the Broadcast LL address
9
10      IF (P IN B) THEN                            # Contending
11          REMOVE P FROM B
12          STOP TIMER
13          DISCARD P
14          RETURN -1                               # Indicates that packet is discarded
15      ELSE                                        # New packet
16          IF ((PV_SE EXISTS) AND (PAI_SE = TRUE)) THEN
17              SET PROG ← (DIST(PV_P, PV_SE) - DIST(PV_P, LPV))
18              IF (PROG > 0) THEN                  # Forwarding progress
19                  ADD P TO B
20                  SET TO                          # Eq. D.1
21                  START TIMER(TO)
22                  RETURN 0                        # Indicates that packet is buffered
23              ELSE
24                  DISCARD P
25                  RETURN -1                       # Indicates that packet is discarded
26              ENDIF
27          ELSE
28              ADD P TO B
29              SET TO ← TO_CBF_MAX
30              RETURN 0                            # Indicates that packet is buffered
31          ENDIF
32      ENDIF
33
34      IF (TIMER(TO) EXPIRES) THEN
35          FETCH P FROM B
36          SET NH_LL_ADDR ← BCAST
37          RETURN NH_LL_ADDR                       # Indicates that packet could be forwarded
38      ENDIF
```

Annex E (normative): GeoBroadcast forwarding algorithms

E.1 Overview

The GeoBroadcast forwarding algorithm is executed by a GeoAdhoc router to relay a packet to the next hop.

The present document defines three forwarding algorithms:

1) Simple GeoBroadcast forwarding algorithm (clause E.2);

2) Contention-based forwarding algorithm for GeoBroadcast (clause E.3);

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

NOTE: For future versions of the present document, forwarding algorithms may still be enhanced. For example, the simple GeoBroadcast forwarding algorithm may be enhanced by calculating the coverage of a certain packet and control redundant re-transmissions.

E.2 Simple GeoBroadcast forwarding algorithm with line forwarding

The algorithm utilizes the function F(x,y) specified in [7] clause 5 in order to determine whether the GeoAdhoc router is located inside, at the border or outside the geographical target area carried in the GeoBroadcast packet header. If the GeoAdhoc router is inside or at the border of the area, the packet shall be re-broadcasted. If it is outside the area, the packet shall be forwarded by the GF algorithm specified in clause D.2.

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

NOTE 2: As defined in [7] clause 5,

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

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 in line forwarding mode no forwarder could be found by the GF algorithm and the packet is buffered in the *UC forwarding packet buffer*,
- -1 indicates that the packet is discarded.

The pseudo-code of the algorithm is below:

```
1     -- P is the GeoNetworking packet to be forwarded
2     -- LAT and LONG are latitude and longitude of the LPV, respectively
3     -- PV_SE is the sender position vector in its LocTE with latitude LAT_SE, longitude LONG_SE
4     --      with LAT_SE and LONG_SE as latitude and longitude
5     --      and position accurancy indicator_PAI_SE
6     -- A is the centre point of the destination area in the GeoNetworking
7     --      packet to be forwarded
8     -- NH_LL_ADDR is the link layer address that identifies the next hop
```

```
9     --      of the GeoNetworking packet
10    -- BCAST is the Broadcast LL address
11    -- GREEDY() is the GF algorithm as specified in clause D.2
12    -- B is the UC forwarding packet buffer
13
14    Calculate F(LAT, LONG)                      # Eq. E.1 in Note 2
15    IF (F ≥ 0) THEN                             # Local GeoAdhoc router is inside or
16                                                # at the border of target area
17        RETURN NH_LL_ADDR ← BCAST
18    ELSE                                        # Local GeoAdhoc router is outside of target area
19        IF ((PV_SE EXISTS) AND (PAI = TRUE)) THEN
20            Calculate F(LAT_SE, LONG_SE)        # Eq. E.1 in Note 2
21            IF (F < 0) THEN                     # Sender is outside of target area
22                RETURN NH_LL_ADDR ← GREEDY(A)   # Greedy() returns LL address of next hop or 0
23            ELSE
24                DISCARD P
25                RETURN -1                       # Indicates that packet is discarded
26            ENDIF
27        ELSE
28            RETURN NH_LL_ADDR ← BCAST
29        ENDIF
30    ENDIF
```

E.3 Contention-based forwarding algorithm for GeoBroadcast

Similar to the contention-based forwarding algorithm for GeoUnicast, with the Contention-based forwarding (CBF) algorithm for GeoBroadcast a receiver decides to be a forwarder of a GBC packet.

When a node broadcasts a GBC packet with the CBF algorithm, all neighbours, which receive the packet, process it, buffer the packet in its CBF packet buffer and start a timer with a timeout that is proportional to the distance between the GeoAdhoc router's local position and the position of the sender, i.e. the node with the maximum forwarding progress will have the smallest timeout (equation E.2). 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 GeoUnicast CBF algorithm, where the distance is the forwarding progress between the GeoAdhoc router's local position and the destination position.

The algorithm also applies line forwarding, i.e. if the local GeoAdhoc router is outside the geographical target area, the packet shall be forwarded by the GF algorithm specified in clause D.2.

Equation E.2: Calculation of timeout TO_CBF_GBC for buffering packets in the CBF packet buffer:

$$TO\_CBF\_GBC = \begin{cases} TO\_CBF\_MAX + \dfrac{TO\_CBF\_MIN - TO\_CBF\_MAX}{DIST\_MAX} \times DIST & \text{for } DIST \leq DIST\_MAX \\ TO\_CBF\_MIN & \text{for } DIST > DIST\_MAX \end{cases} \quad \text{(E.2)}$$

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

TO_CBF_MIN and TO_CBF_MAX shall be set to the GN protocol constants `itsGnGeoBroadcastCbfMinTime` and `itsGnGeoBroadcastCbfMaxTime`, respectively. If DIST_MAX is not defined in the specification of GeoNetworking media-dependent functionality for the specific ITS access technology (e.g. TS 102 636-4-2 [i.2]), it shall be set to the GN protocol constant `itsGnDefaultMaxCommunicationRange`.

Upon expiration of the timer, the GeoAdhoc router re-broadcasts the GBC 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 GBC packet from the CBF packet buffer.

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

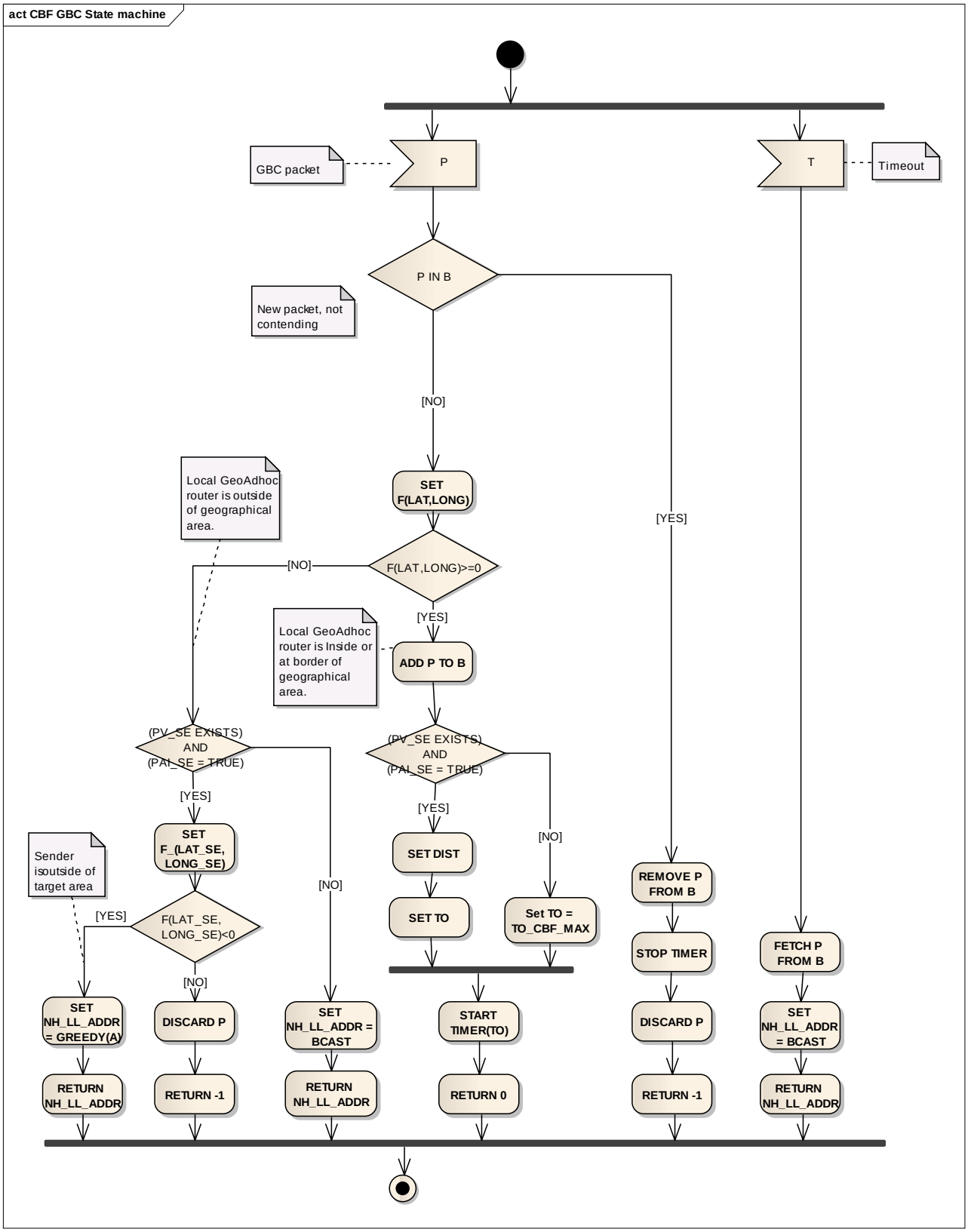

Figure E.1: GeoBroadcast CBF activity diagram

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, either because:
  - in line forwarding mode no forwarder could be found by the GF algorithm and the packet is buffered in the *UC forwarding packet buffer*, or
  - the packet is buffered in the *CBF buffer*,
- -1 indicates that the packet is discarded.

The pseudo-code of the algorithm is below:

```
1     -- P is the GBC packet to be forwarded
2     -- LPV is the local position vector with latitude LAT and longitude LONG
3     -- PV_SE is the sender position vector in its LocTE with latitude LAT_SE and longitude LONG_SE
4     -- B is the CBF packet buffer
5     -- TO is the timeout that triggers the re-broadcast of the packet
6     -- NH_LL_ADDR is the LL address of the next hop
7     -- BCAST is the Broadcast LL address
8
9     IF (P IN B) THEN                                  # Contending
10        REMOVE P FROM B
11        STOP TIMER
12        DISCARD P
13        RETURN -1                                     # Indicates that packet is discarded
14    ELSE                                              # New packet
15        Calculate F(LAT, LONG)                        # Eq. E.1 in Note 2 of clause E.2
16        IF (F ≥ 0) THEN                               # Local GeoAdhoc router is inside
17                                                      # or at the border of target area
18            ADD P TO B
19            IF ((PV_SE EXISTS) AND (PAI = TRUE)) THEN
20                SET DIST ← DIST(PV_SE, LPV)
21                SET TO ← TO_CBF_GBC                   # Eq. E.2 in present clause
22            ELSE
23                SET TO ← TO_CBF_MAX
24            ENDIF
25            START TIMER(TO)
26            RETURN 0                                  # Indicates that packet is buffered
27        ELSE                                          # Local GeoAdhoc router is outside
28                                                      # of target area
29            IF ((PV_SE EXISTS) AND (PAI = TRUE)) THEN
30                Calculate F(LAT_SE, LONG_SE)          # Eq. E.1 in Note 2 of clause E.2
31                IF (F < 0) THEN                       # Sender is outside of target area
32                    RETURN NH_LL_ADDR ← GREEDY(A)     # Greedy() returns LL address of next hop or 0
33                ELSE                                  # Sender is inside or at the border
34                                                      # of target area
35                    DISCARD P
36                    RETURN -1                         # Indicates that packet is discarded
37                ENDIF
38            ELSE
39                SET NH_LL_ADDR ← BCAST
40                RETURN NH_LL_ADDR
41            ENDIF
42        ENDIF
43    ENDIF
44
45    IF (TIMER(TO) EXPIRES) THEN
46        FETCH P FROM B
47        SET NH_LL_ADDR ← BCAST
48        RETURN NH_LL_ADDR
49    ENDIF
```

E.4 Advanced forwarding algorithm for GeoBroadcast

The Advanced forwarding algorithm for GeoBroadcast includes mechanisms from the Greedy Forwarding (GF) algorithm (clause D.2) and the Contention-based forwarding (CBF) algorithm (clause E.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 Advanced forwarding algorithm for GeoBroadcast 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, either because:
  - in line forwarding mode no forwarder could be found by the GF algorithm and the packet is buffered in the *UC forwarding packet buffer*, or
  - the packet is buffered in the *CBF buffer*,
- -1 indicates that the packet is discarded.

At the source, the algorithm checks whether the GeoAdhoc router is located inside or at the border of the geographical target area - if so, it 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 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 by GF (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 GeoAdhoc router 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 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 $\geq$ 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 E.3 and specified in equation E.4 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 E.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.

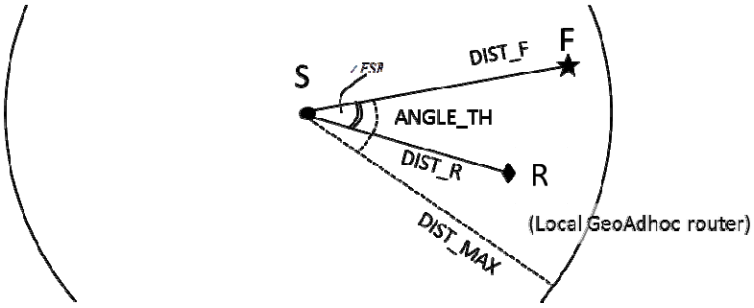

Figure E.2: Sectorial contention area

Equation E.3: Properties of the geometric function G:

$$G = \begin{cases} +1 & \text{inside or at the border of the sectorial area} \\ -1 & \text{outside the sectorial area} \end{cases} \quad \text{(E.3)}$$

Equation E.4: Calculation of the sectorial contention area:

$$G = \begin{cases} +1 & \text{for } (DIST\_R < DIST\_F) \text{ AND } (DIST\_F < DIST\_MAX) \text{ AND } (\angle FSR \leq ANGLE\_TH) \\ -1 & \text{Otherwise} \end{cases} \quad \text{(E.4)}$$

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 example below.

Example:
(1) When DEN_NEIGH < 0,025 node/m^2 → ANGLE_TH = 30°
(2) When 0,025 node/m^2 < DEN_NEIGH < 0,05 node/m^2 → ANGLE_TH = 45°
(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 Advanced GeoBroadcast forwarding algorithm is illustrated in figure E.3.

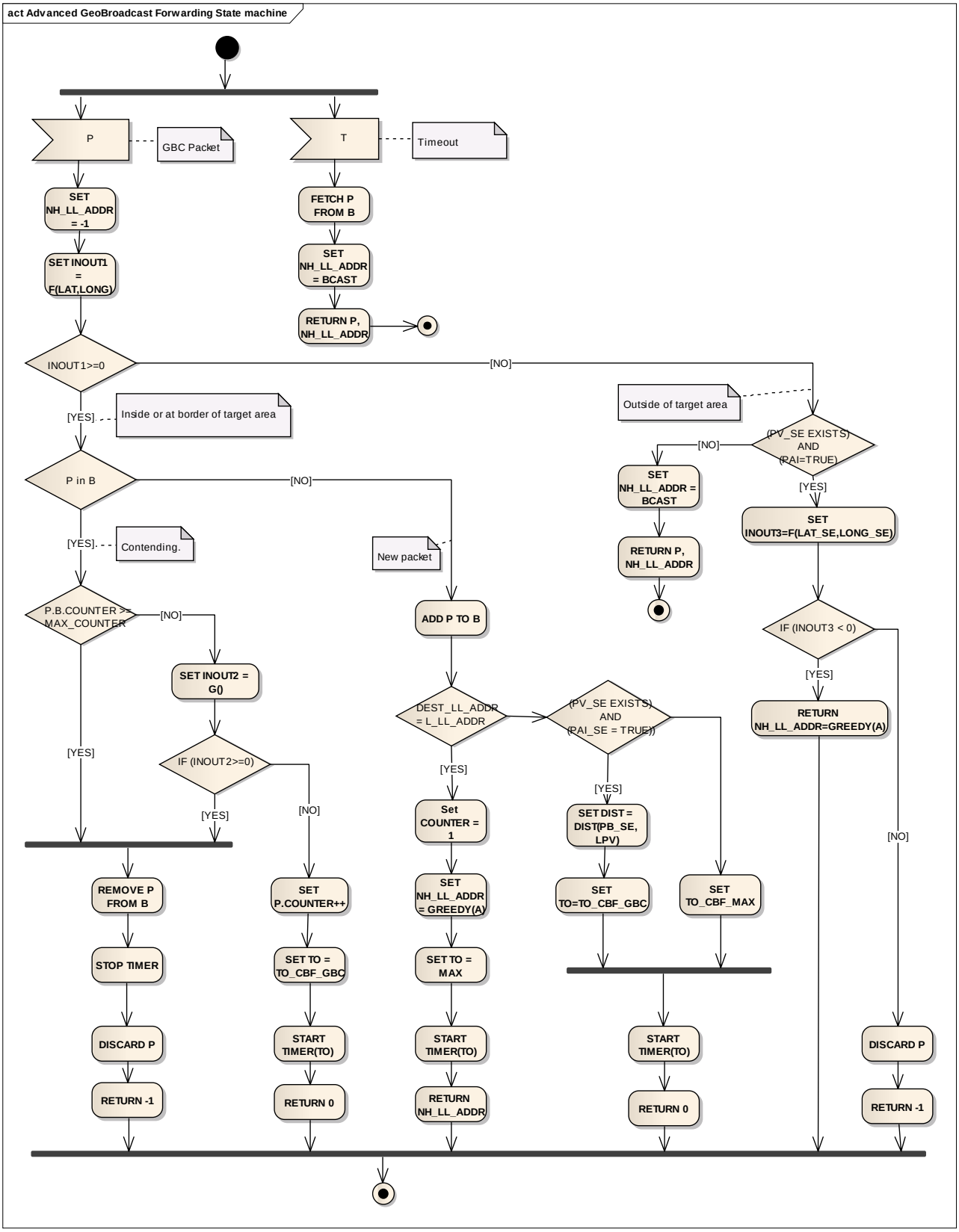

Figure E.3: GeoBroadcast Advanced Forwarding activity diagram

The pseudo-code of the algorithm is below:

```
1     -- P is the GBC packet to be forwarded
2     -- L_LL_ADDR is the LL address of the local GeoAdhoc router
3     -- NH_LL_ADDR is the LL address of the next hop
4     -- DEST_LL_ADDR is the LL destination address carried in P
5     -- B is the CBF packet buffer
6     -- LPV is the local position vector with latitude LAT and longitude LONG
7     -- PV_SE is the sender position vector in its LocTE with latitude LAT_SE, longitude LONG_SE
8     --    and position accuracy indicator PAI
9     -- TO is the timeout that triggers the re-broadcast of the packet
10    -- COUNTER is the retransmit counter for the packet P
11    -- MAX_COUNTER is the retransmit threshold
12    -- BCAST is the Broadcast LL address
13    -- GREEDY() is the GF algorithm as specified in clause D.2
14    -- INOUT1 indicates whether the local GeoAdhoc router is outside the target area or not
15    -- INOUT2 indicates whether the local GeoAdhoc router is outside the sectorial contention
16    --    area or not.
17    -- INOUT3 indicates whether the sender is outside the target area or not
18
19    SET NH_LL_ADDR ← -1                          # Initialize NH_LL_ADDR
20    SET INOUT1 ← F(LAT,LONG)                     # Eq. E.1 in Note 2 of clause E.2
21    IF (INOUT1 ≥ 0) THEN                         # Inside or at border of target area
22        IF (P IN B) THEN                         # Contending
23            IF (B.P.COUNTER ≥ MAX_COUNTER) THEN # Stop contending
24                REMOVE P FROM B                  # Remove packet from CBF buffer
25                STOP TIMER
26                DISCARD P                        # Discard packet
27                RETURN -1                        # Indicates that packet is discarded
28            ELSE
29                SET INOUT2 ← G()                 # Eq. E.4 for sectorial contention area
30                IF (INOUT2 ≥ 0) THEN             # Inside or at the border of sectorial area
31                    REMOVE P FROM B              # Remove packet from CBF buffer
32                    STOP TIMER
33                    DISCARD P                    # Discard packet
34                    RETURN -1                    # Indicates that packet is discarded
35                ELSE                             # Outside of sectorial area
36                    SET P.COUNTER++
37                    SET TO ← TO_CBF_GBC          # Eq. E.2 in clause E.3
38                    START TIMER(TO)
39                    RETURN 0                     # Indicates that packet is buffered
40                ENDIF
41            ENDIF
42        ELSE                                     # New packet
43            ADD P TO B
44            IF (DEST_LL_ADDR = L_LL_ADDR) THEN  # Greedy forwarding
45                SET COUNTER ← 1                  # Initialize COUNTER
46                SET NH_LL_ADDR ← GREEDY(A)       # Greedy()returns LL address of next hop or 0
47                SET TO ← MAX                     # Set to TO_CBF_MAX (Eq. E.2 in clause E.3)
48                START TIMER(TO)
49                RETURN NH_LL_ADDR
50            ELSE                                 # CBF
51                IF ((PV_SE EXISTS) AND (PAI_SE = TRUE)) THEN
52                    SET DIST ← DIST(PV_SE, LPV)
53                    SET TO  ← TO_CBF_GBC         # Eq. E.2 in clause E.3
54                ELSE
55                    SET TO  ← TO_CBF_MAX
56                ENDIF
57                START TIMER(TO)
58                RETURN 0                         # Indicates that packet is buffered
59            ENDIF
60        ENDIF
61    ELSE                                         # Outside of target area
62        IF ((PV_SE EXISTS) AND (PAI_SE = TRUE)) THEN
63            SET INOUT3 ← F(LAT_SE,LONG_SE)          # Eq. E.1 in Note 2 of clause E.2
64            IF (INOUT3 < 0) THEN                    # Sender is outside of target area
65                RETURN NH_LL_ADDR ← GREEDY(A)       # Greedy() returns LL address of next hop or 0
66            ELSE
67                DISCARD P                           # Discard packet
68                RETURN -1                           # Indicates that packet is discarded
69            ENDIF
70        ELSE
71            SET NH_LL_ADDR ← BCAST
72        ENDIF
73    ENDIF
```

```
74    IF (TIMER(TO) EXPIRES) THEN
75        FETCH P FROM B
76        SET NH_LL_ADDR ← BCAST
77        RETURN P, NH_LL_ADDR
78    ENDIF
```

Annex F (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 8.7.5).

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 EN 302 636-4 [i.13], such as for ITS-G5 [i.2].

A mapping between a traffic class to data traffic management mechanisms is configured by the GN management services at start-up (see annex J).

Annex G (normative): GeoNetworking protocol constants

Table G.1 specifies the GeoNetworking protocol constants and their default/initial values.

The protocol constants represent MIB attributes specified in informative annex H.

Table G.1: GeoNetworking protocol constants

Item	GeoNetworking protocol constant	Default/initial value	Comment
1	itsGnLocalGnAddr	1	GeoNetworking address 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 *GN-MGMT* (annex J) ANONYMOUS (2): Local GN_ADDR is configured by the Security entity
3	itsGnProtocolVersion	EN 302 636-4-1 (V1.2.1): 0	Version of the GeoNetworking protocol set in the GeoNetworking protocol headers
4	itsGnStationType	Unknown (0) Pedestrian (1) Cyclist (2) Moped (3) Motorcycle (4) PassengerCar (5) Bus (6) LightTruck (7) HeavyTruck (8) Trailer (9) SpecialVehicles (10) Tram (11) RoadSideUnit (15)	Type of ITS-S
5	itsGnIsMobile	Stationary (0) Mobile (1)	Indicates whether ITS-S is stationary or mobile
6	itsGnIfType	Unspecified (0) ITS-G5 (1)	Indicates type of interface
7	itsGnMinUpdateFrequencyLPV	mobile: 1 000 stationary: 0	Minimum update frequency of local position vector (LPV) in 1/ms
8	itsGnPaiInterval	80	Distance related to the confidence interval for latitude and longitude [m]. Used to determine the PAI (clause 8.5.2).
9	itsGnMaxSduSize	1 398	Maximum size of GN-SDU [octets] 1 500- GN_MAX (88) - GNSEC_MAX (0)
10	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 8.8.2. If the GeoNetworking packet is secured (clause 8.4) the maximum size of the GeoNetworking header is set to the size of the *Basic Header* and the size of the *Secured Packet*.
11	itsGnLifetimeLocTE	20	Lifetime of location table entry [s]

Item	GeoNetworking protocol constant	Default/initial value	Comment
12	itsGnSecurity	DISABLED (0) ENABLED (1)	Indicates whether GN security is enabled (1) or disabled (0).
13	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.
14	itsGnLocationServiceMaxRetrans	10	Maximum number of retransmissions of LS Request packets
15	itsGnLocationServiceRetransmitTimer	1 000	Duration of Location service retransmit timer [ms]
16	itsGnLocationServicePacketBufferSize	1 024	Size of Location service packet buffer [Octets]
17	itsGnBeaconServiceRetransmitTimer	3 000	Duration of Beacon service retransmit timer [ms]
18	itsGnBeaconServiceMaxJitter	itsGnBeaconServiceRetransmitTimer / 4	Maximum beacon jitter [ms]
19	itsGnDefaultHopLimit	10	Default hop limit indicating the maximum number of hops a packet travels
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 GeoAdhoc router [Ko/s]. If the mean (EMA) packet data rate a of a GeoAdhoc router exceeds the value, packets from this GeoAdhoc router (source or sender) are not forwarded.
23	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).
24	itsGnMinPacketRepetitionIntervall	100	Lower limit of the packet repetition interval [ms]
25	itsGnGeoUnicastForwardingAlgorithm	GREEDY (1)	Default GeoUnicast forwarding algorithm
26	itsGnGeoBroadcastForwardingAlgorithm	ADVANCED (3)	Default GeoBroadcast forwarding algorithm
27	itsGnGeoUnicastCbfMinTime	1	Minimum duration a GUC packet shall be buffered in the CBF packet buffer [ms]
28	itsGnGeoUnicastCbfMaxTime	100	Maximum duration a GUC packet shall be buffered in the CBF packet buffer [ms]
29	itsGnGeoBroadcastCbfMinTime	1	Minimum duration a GeoBroadcast packet shall be buffered in the CBF packet buffer [ms]

Item	GeoNetworking protocol constant	Default/initial value	Comment
30	itsGnGeoBroadcastCbfMaxTime	100	Maximum duration a GeoBroadcast packet shall be buffered in the CBF packet buffer [ms]
31	itsGnDefaultMaxCommunicationRange	1 000	Default theoretical maximum communication range [m]
32	itsGnBroadcastCBFDefSectorAngle	30	Default threshold angle for advanced GeoBroadcast algorithm in clause E.4 [degrees]
33	itsGnUnicastCBFDefSectorAngle	30	Default threshold angle for Contention-based forwarding algorithm for GeoUnicast in clause D.3 [degrees]
34	itsGnGeoAreaLineForwarding	ENABLED (1)	Forwarding of GBC/GAC packet if GeoAdhoc router is located outside the GeoArea.
35	itsGnUcForwardingPacketBufferSize	256	Size of UC forwarding packet buffer [Ko]
36	itsGnBcForwardingPacketBufferSize	1 024	Size of BC forwarding packet buffer [Ko]
37	itsGnCbfPacketBufferSize	256	Size of CBF packet buffer [Ko]
38	itsGnDefaultTrafficClass	0x00	Forwarding: Default traffic class

Annex H (informative): ASN.1 encoding of the GeoNetworking MIB

H.1 Use of modules

The ASN.1 module of the present document is **ITSGN {itu-t(0) identified-organization(4) etsi(0) itsgn(2636-4)}** as specified in the following clause.

The ASN.1 module adopts textual conventions defined separately and imported as modules. Objects defined using textual conventions are always encoded by means of the rules that define their primitive type. The adapted subsets of ASN.1 notation described in [i.9] (SNMPv2-SMI) and [i.11] (SNMPv2-TC) are imported.

H.2 ASN.1 module

```
-- ****************************************************************************
-- * ETSI TC ITS EN 302 636-4 GeoNetworking MIB
-- ****************************************************************************

ITSGN-MIB DEFINITIONS::=BEGIN

   IMPORTS
      MODULE-IDENTITY, OBJECT-TYPE,
      Unsigned32, Integer32,
      enterprises                          FROM SNMPv2-SMI
      SnmpAdminString                      FROM SNMP-FRAMEWORK-MIB
      TEXTUAL-CONVENTION, TruthValue       FROM SNMPv2-TC
      InterfaceIndex                       FROM IF-MIB;

-- ****************************************************************************
-- * MODULE IDENTITY
-- ****************************************************************************

itsGn MODULE-IDENTITY
   LAST-UPDATED "201307080000Z"
   ORGANIZATION "ETSI Technical Committee ITS WG3"
   CONTACT-INFO
      "WG Email:    ITS_WG3@LIST.ETSI.ORG"
   DESCRIPTION
      "The MIB module for EN 302 636-4 (GeoNetworking) entities
       itu-t(0).identified-organization(4).etsi(0).itsgn(26364)"
   REVISION     "201307080000Z"
   DESCRIPTION  "EN 302 636-4-1 V1.2.1"
   REVISION     "201307080000Z"
   DESCRIPTION  "EN 302 636-4-1 V1.2.0"
   REVISION     "201306010000Z"
   DESCRIPTION  "EN 302 636-4-1 V0.5.0"
   REVISION     "201305180000Z"
   DESCRIPTION  "EN 302 636-4-1 V0.4.0"
   REVISION     "201302250000Z"
   DESCRIPTION  "EN 302 636-4-1 V0.2.3"
   REVISION     "201301220000Z"
   DESCRIPTION  "EN 302 636-4-1 V0.2.1"
   REVISION     "201206130000Z"
   DESCRIPTION  "Initial version: EN 302 636-4-1 V0.0.2"
::= { enterprises 13019 26364 }

-- ****************************************************************************
-- * PRIMARY GROUPS
-- ****************************************************************************

itsGnObjects     OBJECT IDENTIFIER ::= { itsGn 1 }
itsGnStatistics  OBJECT IDENTIFIER ::= { itsGn 2 }
itsGnConformance OBJECT IDENTIFIER ::= { itsGn 3 }

-- ****************************************************************************
-- * SUB GROUPS
-- ****************************************************************************
```

```
   itsGnMgmt     OBJECT IDENTIFIER ::= { itsGnObjects 1 }

-- ****************************************************************************
-- * SUB SUB GROUPS
-- ****************************************************************************

   itsGnSystem           OBJECT IDENTIFIER ::= { itsGnMgmt 1 }
   itsGnConfig           OBJECT IDENTIFIER ::= { itsGnMgmt 2 }
   itsGnLocationService  OBJECT IDENTIFIER ::= { itsGnMgmt 3 }
   itsGnBeaconService    OBJECT IDENTIFIER ::= { itsGnMgmt 4 }
   itsGnPacketForwarding OBJECT IDENTIFIER ::= { itsGnMgmt 5 }


-- ****************************************************************************
-- * TEXTUAL CONVENTIONS
-- ****************************************************************************

   GnAddress ::= TEXTUAL-CONVENTION
       DISPLAY-HINT "2x:2x:2x:2x"
       STATUS       current
       DESCRIPTION
          "Represents a GeoNetworking address:

           Octets   Contents         Encoding
            1-8     GN address       network-byte order"
       SYNTAX       OCTET STRING (SIZE (8))


-- ****************************************************************************
-- * GN OBJECTS GROUP
-- ****************************************************************************

-- ****************************************************************************
-- * GN SYSTEM GROUP
-- ****************************************************************************

itsGnIfTable        OBJECT-TYPE
    SYNTAX      SEQUENCE OF ItsGnIfEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
    "A table representing the interfaces that will be used by the
    GeoAdhoc router for communication.  Each entry in this table
    represents a configured egress interface.
    "
    ::= { itsGnSystem 1 }


itsGnIfEntry OBJECT-TYPE
    SYNTAX      ItsGnIfEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
    "An entry in the interface table.  It
    represents a single interface entry.
    "
    INDEX  { itsGnIfIndex }
    ::= { itsGnIfTable 1 }

ItsGnIfEntry ::=
    SEQUENCE {
        itsGnIfIndex            InterfaceIndex,
    itsGnIfPriority          Unsigned32,
    itsGnIfDescription       SnmpAdminString
    }

itsGnIfIndex  OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
    "The index of the interface of the GeoAdhoc router.
    "
    ::= { itsGnIfEntry 1 }

itsGnIfPriority   OBJECT-TYPE
    SYNTAX      Unsigned32 (0..255)
```

```
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
    "The priority configured to the interface.
    This value will be configured to a value between 0
    and 255.
    "
    ::= { itsGnIfEntry 2 }

itsGnIfDescription   OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
    "A human-readable textual description of the
    interface on the GeoAdhoc router.
    "
    ::= { itsGnIfEntry 3 }

-- ****************************************************************************
-- * GN CONFIGURATION SUB GROUP
-- ****************************************************************************

itsGnLocalGnAddr OBJECT-TYPE
   SYNTAX      GnAddress
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
       "GeoNetworking address of the GeoAdhoc router."
::= { itsGnConfig 1 }

itsGnLocalAddrConfMethod OBJECT-TYPE
    SYNTAX      INTEGER {
         auto(0),
         managed(1),
     anonymous(2)
     }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
    "AUTO: Local GN_ADDR is configured from MIB
    MANAGED: Local GN_ADDR is configured via the GN management using the service primitive GN-MGMT
(annex J)
    ANONYMOUS: Local GN_ADDR is configured by the security entity"
::= { itsGnConfig 2 }

itsGnProtocolVersion OBJECT-TYPE
   SYNTAX      Unsigned32
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
       "GeoNetworking protocol version."
::= { itsGnConfig 3 }

itsGnStationType OBJECT-TYPE
   SYNTAX      INTEGER{
                  unknown(0),
          pedestrian(1),
          cyclist(2),
          moped(3),
          motorcycle(4),
                  passengerCar(5),
          bus(6),
          lightTruck(7),
          heavyTruck(8),
          trailer(9),
          specialVehicles(10),
          tram(11),
          roadSideUnit(15)
               }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
       "ITS Station type."
::= { itsGnConfig 4 }

itsGnIsMobile OBJECT-TYPE
   SYNTAX      TruthValue
```

```
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Indicates whether ITS station is stationary or mobile."
::= { itsGnConfig 5 }

itsGnIfType OBJECT-TYPE
   SYNTAX      INTEGER{
                  unspecified(0),
          its-g5(1)
               }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
       "ITS interface type."
::= { itsGnConfig 6 }

itsGnMinUpdateFrequencyLPV OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   UNITS       "milliseconds"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Minimum update frequency of local position vector (LPV) in ms."
::= { itsGnConfig 7 }

itsGnPaiInterval OBJECT-TYPE
   SYNTAX      Integer32(0..100)
   UNITS       "meters"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Distance related to the confidence interval of latitude and longitude in m. Used to determine
the Position Accuracy Indicator (PAI)."
::= { itsGnConfig 8 }

itsGnMaxSduSize OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   UNITS       "Octets"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Maximum size of GN-SDU in Octets."
::= { itsGnConfig 9 }

itsGnMaxGeoNetworkingHeaderSize OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   UNITS       "Octets"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Maximum size of GeoNetworking header in Octets."
::= { itsGnConfig 10 }

itsGnLifetimeLocTE OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   UNITS       "Seconds"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Location table maintenance: Lifetime of an entry in the location table
       in s."
::= { itsGnConfig 11 }

itsGnSecurity OBJECT-TYPE
   SYNTAX      INTEGER {
                  disabled   (0),
                  enabled    (1)
               }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Indicates whether GN security is enabled (1) or disabled (0)."
::= { itsGnConfig 12 }

itsGnSnDecapResultHandling OBJECT-TYPE
   SYNTAX      INTEGER {
                  strict     (0),
```

```
                  non-strict (1)
               }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Indicates the handling of the SN-DECAP result code (service
      primitive SN-ENCAP.confirm parameter report). If set to STRICT
      (0), received GN packets that are not correctly verified and
      decrypted (service primitive SN-ENCAP.confirm parameter report
      != CORRECT) are always dropped. If 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."
::= { itsGnConfig 13 }

-- ****************************************************************************
-- * GN LOCATION SERVICE SUB GROUP
-- ****************************************************************************

itsGnLocationServiceMaxRetrans OBJECT-TYPE
   SYNTAX      Integer32(0..255)
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Location service: Maximum number of retransmissions for a LS Request."
::= { itsGnLocationService 1 }

itsGnLocationServiceRetransmitTimer OBJECT-TYPE
   SYNTAX      Integer32(0..65535)
   UNITS       "milliseconds"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Location service: Duration of LS request retransmit timer in ms."
::= { itsGnLocationService 2 }

itsGnLocationServicePacketBufferSize OBJECT-TYPE
   SYNTAX      Integer32(0..65535)
   UNITS       "Octets"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Location service: Size of LS packet buffer in octets."
::= { itsGnLocationService 3 }

-- ****************************************************************************
-- * GN BEACON SERVICE SUB GROUP
-- ****************************************************************************

itsGnBeaconServiceRetransmitTimer OBJECT-TYPE
   SYNTAX      Integer32(0..65535)
   UNITS       "milliseconds"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Beacon service: Duration of Beacon retransmit timer in ms."
::= { itsGnBeaconService 1 }

itsGnBeaconServiceMaxJitter OBJECT-TYPE
   SYNTAX      Integer32(0..65535)
   UNITS       "milliseconds"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Beacon service: Maximum Beacon jitter in ms."
::= { itsGnBeaconService 2 }

-- ****************************************************************************
-- * GN PACKET FORWARDING SUB GROUP
-- ****************************************************************************

itsGnDefaultHopLimit OBJECT-TYPE
   SYNTAX      Integer32(0..255)
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
   "Default hop limit indicating the maximum number of hops a packet travels."
::= { itsGnPacketForwarding 1 }
```

```
itsGnMaxPacketLifetime OBJECT-TYPE
   SYNTAX      Integer32(0..6300)
   UNITS       "seconds"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Upper limit of the maximum lifetime of a packet in s."
::= { itsGnPacketForwarding 2 }

itsGnDefaultPacketLifetime OBJECT-TYPE
   SYNTAX      Integer32(0..6300)
   UNITS       "seconds"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Default value of the maximum lifetime of a packet in s."
::= { itsGnPacketForwarding 3 }

itsGnMaxPacketDataRate OBJECT-TYPE
   SYNTAX      Integer32
   UNITS       "Ko/s"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Maximum packet data rate for a GeoAdhoc router in
      [Ko/s]. If the mean (EMA) packet data rate exceeds the
      value, packets from this GeoAdhoc router (source or sender) are
      not forwarded."
::= { itsGnPacketForwarding 4 }

itsGnMaxGnAreaSize OBJECT-TYPE
   SYNTAX      Integer32
   UNITS       "km^2"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "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 is not sent (source) and not be
      forwarder (forwarder)"
::= { itsGnPacketForwarding 5 }

itsGnMinPacketRepetitionInterval OBJECT-TYPE
   SYNTAX      Integer32(0..1000)
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Lower limit of the packet repetition interval in ms."
::= { itsGnPacketForwarding 6 }

itsGnGeoUnicastForwardingAlgorithm OBJECT-TYPE
   SYNTAX      INTEGER {
                  unspecified   (0),
                  greedy        (1),
                  cbf           (2)
               }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Default GeoUnicast forwarding algorithm."
::= { itsGnPacketForwarding 7 }

itsGnGeoBroadcastForwardingAlgorithm OBJECT-TYPE
   SYNTAX      INTEGER {
                  unspecified   (0),
                  simple        (1),
                  cbf       (2),
          advanced  (3)
               }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Default GeoBroadcast forwarding algorithm."
::= { itsGnPacketForwarding 8 }

itsGnGeoUnicastCbfMinTime OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   MAX-ACCESS  read-only
```

```
   STATUS      current
   DESCRIPTION
      "Minimum duration a GeoUnicast packet is buffered in the CBF packet buffer in ms."
::= { itsGnPacketForwarding 9 }

itsGnGeoUnicastCbfMaxTime OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Maximum duration a GeoUnicast packet is buffered in the CBF packet buffer in ms."
::= { itsGnPacketForwarding 10 }

itsGnGeoBroadcastCbfMinTime OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Minimum duration a GeoBroadcast packet is buffered in the CBF packet buffer in ms."
::= { itsGnPacketForwarding 11 }

itsGnGeoBroadcastCbfMaxTime OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Maximum duration a GeoBroadcast packet is buffered in the CBF packet buffer in ms."
::= { itsGnPacketForwarding 12 }

itsGnDefaultMaxCommunicationRange OBJECT-TYPE
   SYNTAX      Integer32(0..65635)
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Default theoretical maximum communication range in m."
::= { itsGnPacketForwarding 13 }

itsGnGeoAreaLineForwarding OBJECT-TYPE
   SYNTAX      INTEGER {
                  disabled   (0),
                  enabled    (1)
               }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Forwarding of GEOBROADCAST/GEOANYCAST packet if GeoAdhoc
       router is located outside the GeoArea."
::= { itsGnPacketForwarding 14 }

itsGnUcForwardingPacketBufferSize OBJECT-TYPE
   SYNTAX      Integer32(0..255)
   UNITS       "Ko"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Forwarding: Size of UC forwarding packet buffer in Ko."
::= { itsGnPacketForwarding 15 }

itsGnBcForwardingPacketBufferSize OBJECT-TYPE
   SYNTAX      Integer32(0.. 65535)
   UNITS       "Ko"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Forwarding: Size of BC forwarding packet buffer in Ko."
::= { itsGnPacketForwarding 16 }

itsGnCbfPacketBufferSize OBJECT-TYPE
   SYNTAX      Integer32(0.. 65535)
   UNITS       "Ko"
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Forwarding: Size of CBF packet buffer [Ko]."
::= { itsGnPacketForwarding 17 }

itsGnTrafficClass OBJECT-TYPE
   SYNTAX      Integer32(0..255)
```

```
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
      "Forwarding: Default traffic class."
::= { itsGnPacketForwarding 18 }

END
```

Annex I (informative): GeoNetworking data services

I.1 General

The GN data service primitives allow entities of ITS transport protocols to send and receive PDUs via the GN_SAP.

I.2 GN-DATA.request

The service primitive *GN-DATA.request* is used by the ITS transport protocol entity to request sending a GeoNetworking packet. Upon reception of the service primitive *GN-DATA.request*, the GeoNetworking protocol delivers the GeoNetworking packet to the LLC protocol entity via the IN_SAP.

The parameters of the *GN-DATA.request* are as follows:

```
GN-DATA.request (
            Upper protocol entity,
            Packet transport type,
            Destination address,
            Communication profile,
            Security profile, (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).

The *Packet transport type* parameter specifies the packet transport type (GUC, SHB, TSB, GBC, 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 *Communication profile* parameter determines the LL protocol entity (unspecified, ITS-G5).

The *Security profile* parameter determines the security service to invoke.

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, i.e. the T-SDU/GN6-SDU.

I.3 GN-DATA.confirm

The service primitive *GN-DATA.confirm* is used to confirm that the GeoNetworking packet was successfully processed in response to a *GN-DATA.request*. For the reception of the primitive, no behaviour is specified.

The parameters of the service primitive are as follows:

```
GN-DATA.confirm (
            ResultCode
            )
```

The *ResultCode* parameter specifies whether the service primitive *GN-DATA.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 `itsGnMinPacketRepetitionIntervall`;
5) rejected due to unsupported traffic class;
6) rejected due to geographical area exceeds the maximum geographical area size in the GN protocol constant `itsGnMaxGeoAreaSize`; or
7) rejected for unspecified reasons if the service primitive *GN-DATA.request* cannot be accepted for any other reason.

I.4 GN-DATA.indication

The service primitive *GN-DATA.indication* indicates to an upper protocol entity that a GeoNetworking packet has been received. The service primitive is generated by the GeoNetworking protocol 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 *GN-DATA.indication* are as follows:

```
GN-DATA.indication (
            Upper protocol entity,
            Packet transport type,
            Destination, (optional)
            Source position vector,
            Security report, (optional)
            Certificate id, (optional)
            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 *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 *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 J (informative): GeoNetworking management services

J.1 General

The GN management service primitives allow the *ITS Networking & Transport Layer Management* entity to update position, time and GeoNetworking address of the GeoAdhoc router.

J.2 GN-MGMT.request

The service primitive *GN-MGMT.request* is generated by the GeoNetworking protocol at the initialization phase in order to request management information, i.e. time, position vector, GeoNetworking address, TC mapping. After receiving the service primitive *GN-MGMT.request*, the *ITS Networking & Transport Layer Management* entity is in charge of providing the GeoNetworking entity with the requested management information.

The parameter of the *GN-MGMT.request* is as follows:

```
GN-MGMT.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 duplicate address detection or is an initial request.

J.3 GN-MGMT.response

The service primitive *GN-MGMT.response* is generated by the *ITS Networking & Transport Layer Management* entity 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 *GN-MGMT.request* primitive or can be generated unsolicited, i.e. without a service primitive *GN-MGMT.request*.

The parameters of the *GN-MGMT.response* are as follows:

```
GN-MGMT.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 F).

All parameters are optional, whereas at least one parameter is present.

Annex K (informative): Interface to the Security entity

K.1 Security services used by the GeoNetworking protocol

The GeoNetworking protocol may exchange information with the security entity via the Sec_GN_SAP (figure 1). The Sec_GN_SAP may be realized as SN-SAP [i.4].

The GeoNetworking protocol may use the following security services [i.4]:

1) SN-ENCAP (clause K.2),
2) SN-DECAP (clause K.3),
3) SN-IDCHANGE-SUBSCRIBE (clause K.4),
4) SN-IDCHANGE-UNSUBSCRIBE (clause K.6).

K.2 SN-ENCAP service

As specified in [i.4], the SN-SIGN service provides the service primitives *SN-ENCAP.request* and *SN-ENCAP.confirm*.

The service primitive *SN-ENCAP.request* is sent from the GeoNetworking protocol to the security entity for executing the ENCAP service. The minimum data being passed from the GeoNetworking protocol to the security entity are specified in table K.1.

Table K.1: SN-ENCAP service: Data passed from the GeoNetworking protocol to the security entity

Data	Data requirement	Mandatory/Optional
tbe_packet_length	Length of the packet to encapsulate into the security envelope.	Mandatory
tbe_packet	The packet to be encapsulated into the security envelope.	Mandatory
sec_profile	Identifier of the security profile to determine the security service to invoke.	Optional

The service primitive *SN-ENCAP.confirm* is sent from the security entity to the GeoNetworking protocol as a corresponding reply to a *SN-ENCAP.request*. The minimum data being passed from the security entity to the GeoNetworking protocol are specified in table K.2.

Table K.2: SN-ENCAP service: Data passed from the security entity to the GeoNetworking protocol

Data	Data requirement	Mandatory/Optional
sec_packet_length	Length of the *Secured Packet* [octets].	Mandatory
sec_packet	The *Secured Packet*.	Mandatory

K.3 SN-DECAP service

As specified in [i.4], the SN-DECAP service provides the service primitives *SN-DECAP.request* and *SN-DECAP.confirm*.

The service primitive *SN-DECAP.request* is sent from the GeoNetworking protocol to the security entity for executing the SN-DECAP service. The minimum data being passed from the GeoNetworking protocol to the security entity are specified in table K.3.

Table K.3: SN-DECAP service: Data passed from the GeoNetworking protocol to the security entity

Data	Data requirement	Mandatory/Optional
sec_packet_length	Length of the *Secured Packet* [octets].	Mandatory
sec_packet	Octet string containing the *Secured Packet*.	Mandatory

The service primitive *SN-DECAP.confirm* is sent from the security entity to the GeoNetworking protocol as a corresponding reply to a *SN-DECAP.request*. The minimum data being passed from the security entity to the GeoNetworking protocol are specified in table K.4.

Table K.4: SN-DECAP service: Data passed from the security entity to the GeoNetworking protocol

Data	Data requirement	Mandatory/Optional
plaintext_packet_length	Length of the decrypted and verified packet.	Mandatory
plaintext_packet	The decrypted and verified packet.	Mandatory
report	Verify and decrypt result flags, to be forwarded to ITS Facilities layer: SUCCESS FALSE_SIGNATURE INVALID_CERTIFICATE REVOKED_CERTIFICATE INCONSISTENT_CHAIN INVALID_TIMESTAMP DUPLICATE_MESSAGE INVALID_MOBILITY_DATA UNSIGNED_MESSAGE SIGNER_CERTIFICATE_NOT_FOUND UNSUPPORTED_SIGNER_IDENTIFIER_TYPE INCOMPATIBLE_PROTOCOL UNENCRYPTED_MESSAGE DECRYPTION_ERROR	Mandatory
certificate_id	Identification of the source certificate, e.g. by the certificate hash, to be forwarded to facilities.	Optional
permissions	In case the used security protocol is capable of attaching the senders permissions, the SN-DECAP service may report those back to the caller.	Optional

K.4 SN-IDCHANGE-SUBSCRIBE service

As specified in [i.4], the IDCHANGE-SUBSCRIBE service provides the service primitives *SN-IDCHANGE-SUBSCRIBE.request* and *SN-IDCHANGE-SUBSCRIBE.confirm*.

The service primitive *SN-IDCHANGE-SUBSCRIBE.request* is sent from the GeoNetworking protocol to the security entity for executing the *IDCHANGE-SUBSCRIBE* service. The minimum data being passed from the GeoNetworking protocol to the security entity are specified in table K.5.

Table K.5: IDCHANGE-SUBSCRIBE service:
Data passed from the GeoNetworking protocol to the security entity

Data	Data requirement	Mandatory/Optional
idchange_event_hook	Callback function which is called when a id-change event occurs.	Mandatory
subscriber_data	Parameter for the callback function	Optional

The service primitive *SN-IDCHANGE-SUBSCRIBE.confirm* is sent from the security entity to the GeoNetworking protocol as a corresponding reply to a *SN-IDCHANGE-SUBSCRIBE.request*. The minimum data being passed from the security entity to the GeoNetworking protocol are specified in table K.6.

**Table K.6: IDCHANGE-SUBSCRIBE service:
Data passed from the security entity to the GeoNetworking protocol**

Data	Data requirement	Mandatory/Optional
subscription	Subscription handle for unsubscribe.	Mandatory

K.5 SN-IDCHANGE-EVENT service

As specified in [i.4], the IDCHANGE-EVENT service provides the service primitives *SN-IDCHANGE-EVENT.indication* and *SN-IDCHANGE-EVENT.response*.

The service primitive *SN-IDCHANGE-EVENT.indication* is sent from the security entity to the GeoNetworking protocol for executing the *IDCHANGE-EVENT* service. The minimum data being passed from the security entity to the GeoNetworking protocol are specified in table K.7.

**Table K.7: IDCHANGE-EVENT service:
Data passed from the security entity to the GeoNetworking protocol**

Data	Data requirement	Mandatory/Optional
command	Id change phase (PREPARE, COMMIT, ABORT, DEREG)	Mandatory
Id	Id to be set.	Mandatory
subscriber_data	Additional parameter for callback function internal use. This will be passed to the hook function on every call.	Optional

The service primitive *SN-IDCHANGE-EVENT.response* is sent from the GeoNetworking protocol to the security entity as a corresponding reply to a *SN-IDCHANGE-EVENT.response*. The minimum data being passed from the GeoNetworking protocol to the security entity are specified in table K.8.

**Table K.8: IDCHANGE-EVENT service:
Data passed from the GeoNetworking protocol to the security entity**

Data	Data requirement	Mandatory/Optional
return_code	Acknowledgement to the given command (TRUE, FALSE).	Mandatory

K.6 SN-IDCHANGE-UNSUBSCRIBE service

As specified in [i.3], the IDCHANGE-UNSUBSCRIBE service provides the service primitives *SN-IDCHANGE-UNSUBSCRIBE.request* and *SN-IDCHANGE-UNSUBSCRIBE.confirm*.

The service primitive *SN-IDCHANGE-UNSUBSCRIBE.request* is sent from the GeoNetworking protocol to the security entity for executing the *IDCHANGE-UNSUBSCRIBE* service. The minimum data being passed from the GeoNetworking protocol to the security entity are specified in table K.9.

**Table K.9: IDCHANGE-UNSUBSCRIBE service:
Data passed from the GeoNetworking protocol to the security entity**

Data	Data requirement	Mandatory/Optional
subscription	Subscription handle, given through subscribe	Mandatory

The service primitive *SN-IDCHANGE-UNSUBSCRIBE.confirm* is sent from the security entity to the GeoNetworking protocol as a corresponding reply to a *SN-IDCHANGE-UNSUBSCRIBE.request*. The minimum data being passed from the security entity to the GeoNetworking protocol are specified in table K.10.

**Table K.10: IDCHANGE-UNSUBSCRIBE service:
Data passed from the security entity to the GeoNetworking protocol**

Data	Data requirement	Mandatory/Optional
(none)		

Annex L (informative): Bibliography

ETSI TS 102 890-1: "Intelligent Transport Systems (ITS); Facilities layer function; Communication Management specification".

ETSI TS 102 890-2: "Intelligent Transport Systems (ITS); Facilities layer function; Services announcement 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"Intelligent Transport Systems; OSI cross-layer topics; Part 1: Architecture and addressing schemes".

ETSI TS 102 723-4: "Intelligent Transport Systems; OSI cross-layer topics; Part 4: Interface between management entity and network and transport layers".

ETSI TS 102 723-10: "Intelligent Transport Systems; OSI cross-layer topics; Part 10: Interface between access layer and network and transport layers".

ETSI TS 102 723-11: "Intelligent Transport Systems; OSI cross-layer topics; Part 11: Interface between network and transport layers and facilities layer".

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.

History

Document history		
V1.1.1	June 2011	Publication as TS 102 636-4-1
V1.2.0	October 2013	EN Approval Procedure AP 20140201: 2013-10-04 to 2014-02-03