



Geographic addressing and routing
for vehicular communications
<http://www.geonet-project.eu/>



ICT-2007.6.1: ICT for intelligent
vehicles and mobility services

GeoNet STREP N°216269

D6.1 GeoNet Porting and Integration Guideline

DATE	3 rd March 2010
CONTRACTUAL DATE OF DELIVERY TO THE EC	M24 – January 2010
ACTUAL DATE OF DELIVERY TO THE EC	M25 – February 2010
EDITOR, COMPANY	Wilfried Lohmann & Carsten Schulze, lesswire
WORKPACKAGE	WP6 Integration and porting
DOCUMENT CODE	GeoNet-D6.1-v1.0
SECURITY	Public

DOCUMENT HISTORY

Release	Date	Reason of change	Status	Distribution
0.1	23/11/09	Document structure	Draft	Internal
0.2	11/01/10	Updated document structure considering GeoNet partner remarks	Draft	Internal
0.3	18/01/10	First internal draft for internal discussion	Draft	Internal
0.4	18/01/10	Updated cross link references	Draft	Internal
0.41	21/01/10	Carsten's comments considered	Draft	Internal
0.56	02/04/10	Updated sections 4 and 5	Draft	Internal
0.57	02/05/10	Updated section 6	Draft	Internal
0.58	02/07/10	Updated section 4 and 5	Draft	Internal
0.59	11/02/10	Comments of INRIA(Thierry)	Draft	Internal
0.60	12/02/10	Update according to comments	Draft	Internal
0.61	24/02/10	Included comments of GeoNet partners	Draft	Internal
0.9	26/02/10	Pre-release. Text formatted	Draft	Internal
1.0	03/03/10	Final	Final	EC

Name of the coordinating person: Arnaud de La Fortelle, INRIA
E-mail: Arnaud.de_La_Fortelle@inria.fr

Contents

1. Executive Summary	4
2. Structure of the Document	6
3. System overview	7
3.1 GeoNet as part of ITS architecture	7
3.2 Protocol stack overview	10
3.3 Porting and integration goals	12
3.4 Further information in related GeoNet documents	13
4. Porting and integration	14
4.1 Target system requirements	14
4.1.1 Target hardware	14
4.1.2 Software and tool requirements	15
4.1.3 Development platform	17
4.2 Porting and integration of GeoNet modules	17
4.2.1 Platform independence of GeoNet stack	17
4.2.2 Management Layer	19
4.2.3 Lower Layer	21
4.2.4 C2CNet Layer	23
4.2.5 IPv6 Layer	24
4.2.6 Upper layers	26
4.2.7 Tools for testing	26
4.3 GeoNet Installation and User Guide	27
5. CVIS Porting example	28
5.1 Lightweight integration	28
5.2 Full GeoNet integration	30
5.2.1 Modules to port	30
5.3 Implementation on CVIS platform	31
5.3.1 Lightweight GeoNet integration to support OSGi GeoNet aware applications	31
5.3.2 Full integration	32
5.3.3 Evaluation of full integration	34
5.4 Porting and integration results - outlook	34
6. Conformance testing	35
6.1 Benefit of compliance tests	35
6.2 Conformance test basic understandings	35
6.3 Test environment and test step sequence	36
6.4 Test results	37
7. License policy and porting support	38
7.1 Management of IPR and license policy	38
7.2 Getting support	40
Annex A: Terminology & Acronyms	42
Annex B: Contributors	44
Annex C: Getting support for porting and integration	45

Annex D: References.....	46
---------------------------------	-----------

1. Executive Summary

The GeoNet project integrates IPv6 networking and Car-to-Car Communication Consortium's (C2C-CC) GeoNetworking capabilities to one single protocol stack for Intelligent Transportation Systems (ITS) and references it as *IPv6 GeoNetworking*.

Both the architecture and specification are described in [GeoNetD1.2] and [GeoNetD2.2], respectively. Two independent prototype implementations of the C2CNet layers are provided and integrated in a multicast-enabled and UMIP-enabled IPv6 stack with some extensions for location privacy (pseudonyms) and direct V2V communications (MNPP) [GeoNetD3.1]. A TTCN-3 based conformance test tool is available [GeoNetD4.1] to prove the compliance of implementations with respect to the specification [GeoNetD2.2]. In order to disseminate project results, individual GeoNet partner submitted detailed reference specifications to standardisation bodies (IETF, ETSI, ISO) (see [GeoNetD8.3]).

The purpose of this document is to guide interested users into porting the GeoNet protocol stack to their target system since one of the GeoNet main goals is to make the project results available to other EU and national ITS projects. As a porting example the GeoNet protocol stack is specified and partly ported to the target system of one of the supporting projects – the EU funded CVIS project (<http://www.cvisproject.org/>) which has itself developed around the CALM architecture (ISO TC204 WG16) [ISO-21217].

An in-depth description of modules and interface implementation of GeoNet protocol stack is provided in deliverable “D3.1.Development Results” ([GeoNetD3.1]). The present porting guide will not repeat the description delivered in D3.1 since Annexes A to G of [GeoNetD3.1] give a very detailed picture of the implementation. Instead this porting guide describes the critical and system dependent parts. This guide should allow third parties to get an understanding of the work needed to integrate the GeoNet stack into their own ITS environment and to identify the pitfalls. Note however that [GeoNetD3.1] is not publicly available so the interested reader is encouraged to look for more information in [GeoNetD8.3] and to contact GeoNet partners as indicated in Annex C of the present document. GeoNet will deliver [GeoNetD3.1] to requesting third parties after reaching a non-disclosure agreement (NDA) .

Due to available resources of partners within GeoNet the Porting and Integration Guide concentrates on Linux based systems. Nevertheless porting to other e.g. automotive grade operating systems is possible without big effort if IPv6 is supported on those platforms and a ETSI ITS-G5 or IEEE 802.11p hardware interface is available.

In order to give an overview, the scope of GeoNet within the C2C-CC architecture is outlined in Figure 1. Please note that GeoNet explicitly supports the “MAC/LLC, C2C MAC Layer Extension, European IEEE 802.11p” and “PHY, European IEEE 802.11p” depicted in Figure 1. These extensions are supported as ETSI ITS-G5 compliant devices.

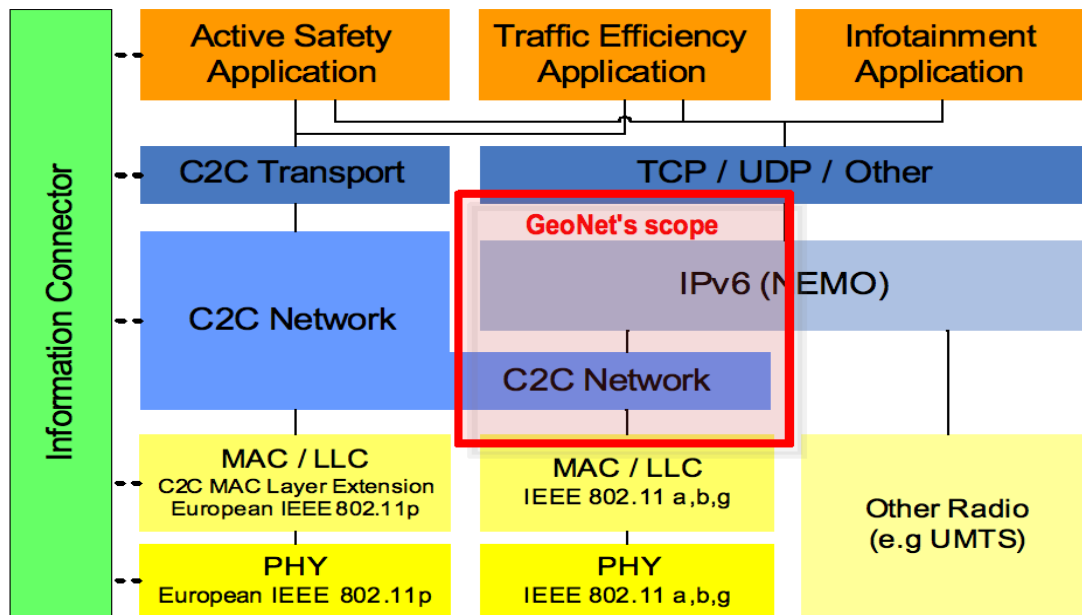


Figure 1: C2C-CC architecture and scope of GeoNet [GeoNetD1.2]

2. Structure of the Document

The present document is structured as follows:

- Section 3 “System overview” provides a short system overview including the related GeoNet documents for easy use of this Porting Guideline. Functional modules and SAPs subject to porting are over viewed.
- Section 4 “Porting and integration” describes the integration and porting process. It provides the minimal and typical system requirements of the target system and gives important additional information for porting and integration.
- Section 5 “CVIS Porting example” provides an example of integration of GeoNet into European ITS architecture and specifies the interfaces needed for a full integration of GeoNet into CALM. CVIS is preferred as a target platform because it follows the CALM standard.
- Section 6 “Conformance testing” provides information on conformance testing according to the specification in [GeoNetD2.2].
- Section 7 “License policy and porting support” points out the management of IPR and license policy of GeoNet. It shows what parts of the stack are publicly available and what parts are subject to individual license. This section also directs the reader to GeoNet project partners in order to get additional information or help for integration and porting.
- Annex A: Terminology & Acronyms describes the specific terminology as used in this document. Note that the GeoNet common terminology and acronyms are provided in [GeoNetD1.2], In Annex A are only listed additional terms and acronyms used within this deliverable and not defined in [GeoNetD1.2].
- Annex B: Contributors names the authors contributed to this document
- Annex C: Getting support for porting and integration gives the contact addresses of GeoNet partners to get porting and integration support.
- Annex D: References lists all the references provided in this document

3. System overview

In order to port the GeoNet protocol stack and integrate it into a third party platform, the software engineer should have some basic knowledge about the subject of porting. The intention of this section is to provide basic information about GeoNet modules, their functions and the interfaces in between. Additionally links to further information in GeoNet documents are provided.

3.1 GeoNet as part of ITS architecture

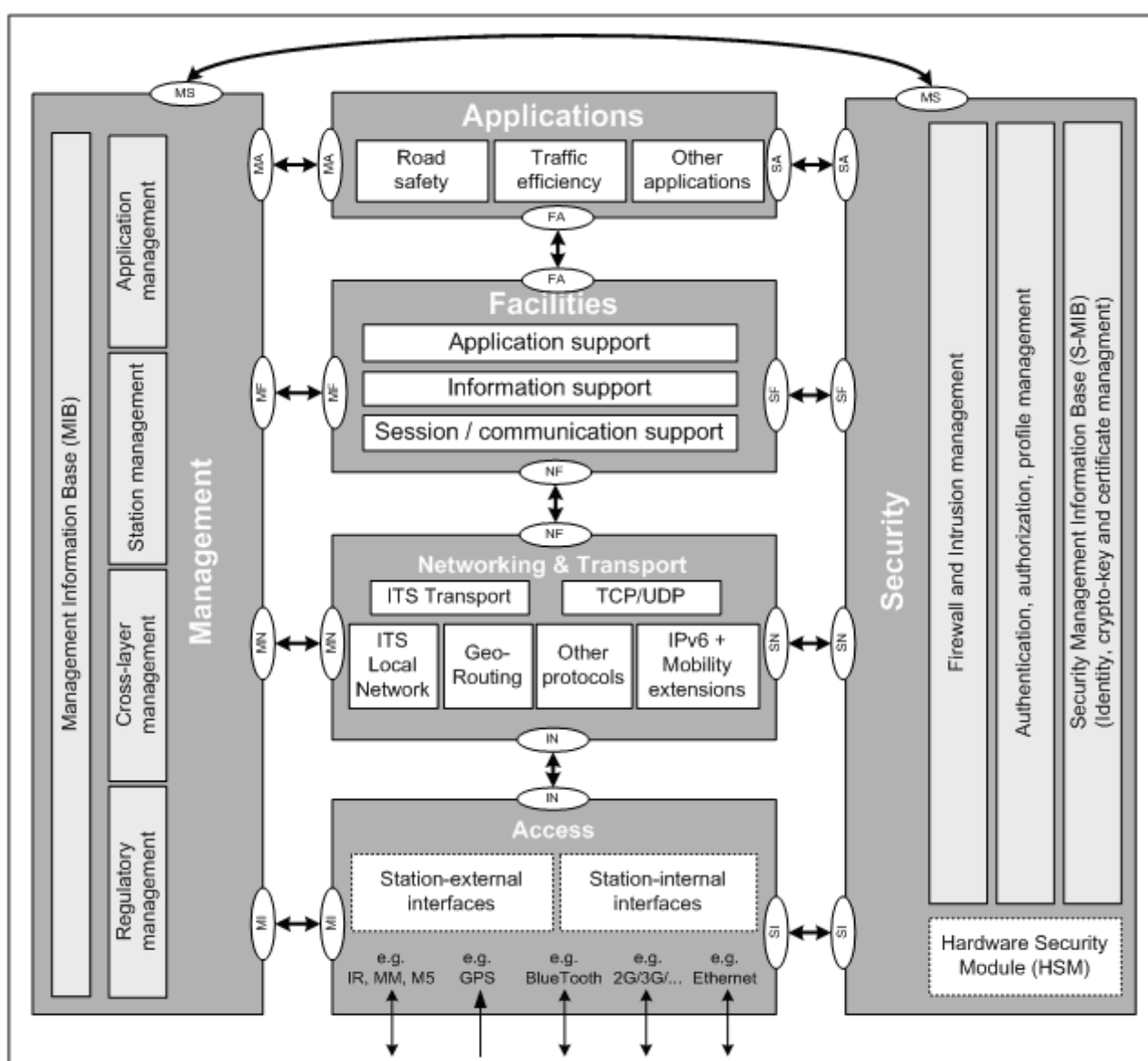


Figure 2: ITS architecture [ETSI-TS-102-665]

In Figure 2 the ETSI ITS architecture is provided. In layer “*Networking & Transport*” a GeoRouting module is foreseen. However, Figure 2 does not display the link between IPv6 and GeoNetworking.

Figure 3 provides a detailed view of Network and Transport layer of the ITS station reference architecture. Note, the ITS Transport and TCP/UDP modules are not shown in this figure. The GeoNet project team worked out together with colleagues of ETSI ITS WG3, CVIS and ISO TC204 WG16 a proposal to present the GeoNet protocol stack within the ITS station reference architecture, see [GeoNet-Integration].

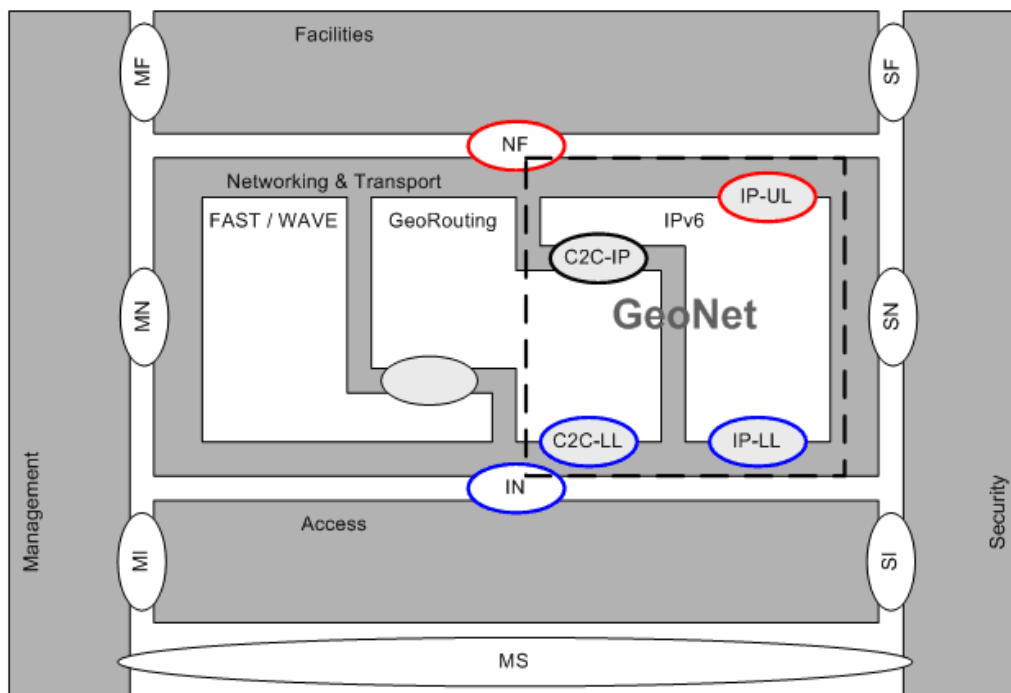


Figure 3: Proposal for integration of GeoNet into ITS Networking & Transport layer of ITS station architecture [GeoNet-Integration]. Note: ITS Transport /TCP/UDP modules are not shown in this drawing.

In Figure 3 the IPv6 layer shape is slightly adapted¹ to ease the understanding: Both the SAP C2C-LL (2A: Egress Interface) and the SAP IP-LL (2B: Ingress Interface) are linking the GeoNet stack to the Logical Link Control sub layer. C2CNet is a part of the ETSI ITS GeoRouting module linked to the IPv6 layer through SAP C2C-IP. Note, the management cross (or vertical) layer and the associated SAPs are not shown in Figure 3 for the sake of clarity. However, the SAPs currently defined by GeoNet fit into the ITS SAP scheme. For standardisation, an ASN.1 specification of the service primitives is needed (alternatively XML). Existence of an ASN.1 specification does not imply that these service primitives are testable. The GeoNet SAPs may be mapped to ITS SAPs as shown in Table 1.

¹ In contrast to Figure 4

ITS SAP according to [ETSI-TS-102-665]	Mapped GeoNet SAP	Remarks
MI	MNG-LL	SAP between Management Layer and Logical Link sub-layer (Module 2A: Egress Interface). SAP MNG-LL is currently not specified as a standalone SAP and is not shown in Figure 3.
MN	MNG-C2C	SAP between Management Layer and C2CNet Layer (not shown in Figure 3).
	MNG-IP	SAP between Management Layer and IPv6 Layer (not shown in Figure 3).
MF	MNG-UL	SAP between Management Layer and upper layers like OSI Transport or ITS facility layers. (not shown in Figure 3).
SF	MNG-C2C	Security related part of SAP between Management Layer and C2CNet Layer (not shown in Figure 3).
	MNG-IP	Security related part of SAP between Management Layer and IPv6 Layer (not shown in Figure 3).
SF	MNG-UL	Security related part of SAP between Management Layer and upper layers like OSI Transport or ITS facility layers. (not shown in Figure 3).
IN	C2C-LL	SAP between C2CNet module and OSI Data Link Layer or ITS Access Layer
	IP-LL	SAP between IPv6 module and OSI Data Link Layer or ITS Access Layer
NF	IP-UL	SAP between IPv6 module and upper layers like OSI Transport or ITS facility layers.

Table 1: Mapping of GeoNet SAPs to SAPs of European ITS architecture

The proposed integration of the GeoNet functionality into the ITS station reference architecture will be intensively discussed in order to prepare and enhance standardisation

in SDOs (Standard Developing Organisations) like ETSI TC ITS WG 3 or ISO TC204 WG16 (CALM).

3.2 Protocol stack overview

Figure 4 provides an overview of main GeoNet functional modules that are subject to

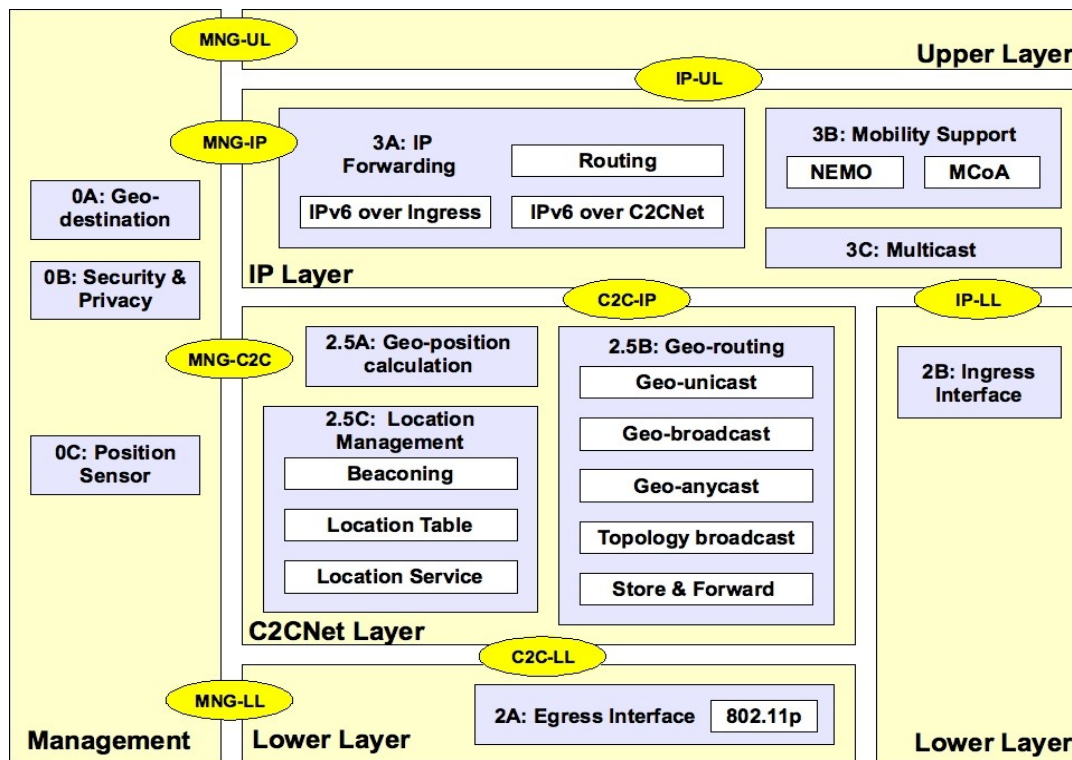


Figure 4: Main Functional Modules [GeoNetD1.2]

porting and integration work. A short overview of functionalities needed for porting and integration is given below.

IP Layer: In current prototype implementations the IP Layer is a state of the art Linux IPv6 stack. Additionally some of the standard modules are extended by GeoNet or newly developed:

- **Module 3A: IP Forwarding:** This module performs router functions on MR and AR. It acquires necessary IP parameters and routes IPv6 packets over the Service Access Points (SAP) IP-UL, C2C-IP and IP-LL to their destination or next hop, respectively.
- **Module 3B: Mobility Support:** 3B maintains Internet connectivity and provides session continuity to the GeoNet MR. It is present on MR only.

- **Module 3C: Multicast:** The Multicast module acquires group membership information, it determines if there are listeners on its interfaces for a given multicast group and feeds module “3A: IPv6 Forwarding” with necessary information to update the routing table.

C2CNet Layer: The modules of C2CNet layer are located in the ITS Station networking layer. These modules are the core modules of GeoNet stack and responsible for GeoNetworking.

- **Module 2.5A: Geo-position calculation:** Calculates the position information and accomplishes the geographical relevance check.
- **Module 2.5B: Geo-routing:** The module is in charge of forwarding packets based on geographical information such as position or velocity.
- **Module 2.5C: Location management:** The module manages location information by acquiring neighbourhood information from other C2CNet nodes such as position or velocity, records it locally and resolves problems when no position information is available.

Lower Layer: These modules provide a platform independent interface to the GeoNet stack. However, the implementations of these modules themselves are platform dependent.

- **Module 2A Egress Interface:** This module provides access to the physical network interface for communicating with other GeoNet OBUs and RSUs. Only ETSI ITS G5 and IEEE 802.11p are considered so far in the scope of the GeoNet project. However, other media could be supported by adaptation of this module.
- **Module 2B Ingress Interface:** This interface links the GeoNet protocol stack to other nodes on the same subnet. Note, that for well-known ingress interfaces such as Ethernet or IEEE 802.11 existing specification can be used without modification. No further implementation work is required, but standard behaviour is expected. Any technology such as Ethernet or State-of-the-Art LLC implementations can be used. The specification of this module is thus out of scope of the GeoNet project. This module communicates with the IP layer through SAP IP-LL.

Management Layer: This cross-layer (or vertical) plane comprises typical management-level functions, e.g. to provide position and information about the target geographic destination area (GeoDestination) to modules on different layers or take care of security and privacy. That requires a strong cooperation between different layer modules.

- **Module 0A: GeoDestination:** This module allows to exchange information of GeoDestination between application layer and C2CNet layer. Consequently, this module allows to map a well-known multicast address to a target GeoDestination (latitude, longitude, radius, etc.). This mapping is recorded in a table and accessed by both layers or encoded in the multicast address itself, but still requires the share

of knowledge between the layers, see [GeoNetD1.2], sub-clauses 6.2.1 and 6.7.6 for details.

- **Module 0B: Security & Privacy:** This module is in charge of changing the C2CNet ID and the associated IPv6 address bound to this C2CNet ID. In that case the geographic location of the vehicle cannot be revealed from the IPv6 address carried in the IPv6 header. Note, there is more information available in [GeoNetD1.2], sub-clause 6.2.2, 6.7, and Annex B.
- **Module 0C: Position Sensor:** The Position Sensor module provides geographic information to modules in the C2CNet Layer and the Upper Layer.

Upper Layer: The upper layer includes the transport layer and the application layer. There are no specific GeoNet protocol stack modules in upper layers. However, applications need to be Geo-aware in order to transmit packets to a given geographic area (the GeoDestination information must be encoded at the IPv6 layer using one of the means described in [GeoNetD2.2]). Applications or tools may access the GeoNet stack by SAPs IP-UL and MNG-UL for data transfer and management tasks, respectively.

3.3 Porting and integration goals

This sub-clause presents the goals which have led the GeoNet partners to create a porting and integration guideline in its present form. These goals take into account the motivations behind WP6 namely to support the easy use of project results within other EU and national ITS projects and to contribute to European and international standardisation. The porting and integration goals are as follows:

1. **Check proper functioning on other platforms** than the GeoNet prototype implementations. This is a proof of concept porting to a platform of one of the supporting EU projects.
2. **Prove platform independence of GeoNet specifications and implementation:** Provide an integrated portable version of one of the developed prototype implementations with clearly identified parts of source code that is platform dependent.
3. **Provide required information to potential users** by bundling the knowledge needed to deploy GeoNet project results. Porting to CVIS is done to check and improve the *Porting and Integration Guide based on GeoNet's own experience on porting and integration*.

Note that this document identifies the critical and system dependent tasks of porting and integration. A very detailed description of module and interface implementation of GeoNet stack is provided in the annexes of deliverable “D3.1 Development Results”, see [GeoNetD3.1].

3.4 Further information in related GeoNet documents

The GeoNet project has contributed the following public or confidential deliverables. All public deliverables are available for download on the GeoNet web site (<http://www.geonet-project.eu/>). Please note that confidential documents are by definition not publicly available:

- The GeoNet architecture and specification are provided in public documents [GeoNetD1.2] (Architecture) and [GeoNetD2.2] (Specification), respectively.
- Two independent prototype implementations of the C2CNet layer, the implementation of IPv6 over C2CNet adaptation sub-module at the IPv6 layer and their integration into an IPv6 GeoNetworking protocol stack are described in [GeoNetD3.1] (Implementation). Note that the binary files and many of the source code modules as provided by [GeoNet6.2] (Software Release) are made available on a *confidential basis*.
- TTCN-3 based conformance tests described in [GeoNetD4.1] have been defined to prove the compliance of GeoNet implementation with the specification [GeoNetD2.2]. This conformance test tool may be used by 3rd parties to test their own implementations.
- The emulation environment and results of performance evaluation are described in [GeoNetD5.1].
- This deliverable D6.1 provides a *Porting and Integration Guideline* to support interested 3rd parties to port the GeoNet stack to other platforms. Note that deliverable [GeoNetD6.2] "*Final GeoNet integrated prototype implementation (Software)*" provides a CD with software and documentation as described in this document.
- The deliverable [GeoNetD7.1] makes available the evaluation results of tests and performance evaluation conducted indoor and with real vehicles.
- In order to disseminate project result individual GeoNet partner submitted detailed reference specifications to standardisation bodies (IETF, ETSI, ISO). There are many other dissemination activities as well.

4. Porting and integration

Since within the context of the project not all possible target systems can be addressed a common porting and integration guideline is developed. This section provides a common overview of porting and integration issues. In contrast, Section 5. CVIS Porting example delivers information about two very different approaches of porting to CVIS EU project, thus applying the basic principles provided in this section.

4.1 Target system requirements

4.1.1 Target hardware

Minimum requirements are as follows:

- CPU Typical performance 180 MIPS² (comparable to ARM9@180MHz)
- Flash 64MB
- SDRAM 64MB
- Position sensor (e.g. GPS device, position update rate 2Hz)
- Egress interface: One ETSI ITS-G5A³ or IEEE802.11p compatible NIC (network interface card).
- Ingress interface: Ethernet NIC @ 10Mbps

Typical hardware platform consists of following components:

- CPU Typically performance > 750 MIPS (e.g. XScale/ARM11 core / 64bits MIPS@ 266MHz)
- Flash 128MB NAND-Flash and 128MB SDRAM
- Position sensor (e.g. GPS device with update rate of 4Hz), gyrometer, odometer to provide more accurate position information or update position information when GPS is not available

² MIPS – Million Instructions Per Second

³ [ETSI-ES-202-663] in V1.1.0 (2010-01) requires that ITS-G5 STAs (stations) operating on both the G5CC (ITS-G5 Control Channel) and one of the G5SCs (ITS-G5 Service Channels) have to be able to simultaneously receive on both channels except whilst transmitting in any one of these channels.

- Egress interface: One or two ETSI ITS-G5A or IEEE802.11p compatible NIC, antenna diversity
- Ingress interface: Ethernet NIC @ 10/100Mbps

Note: Typically such a device is equipped by an interface to a vehicle bus like CAN, MOST, or similar. However, such an interface is not required for proper function of GeoNet protocol stack. Other ITS related functions may be implemented and run in parallel on such a target platform. In principle there are no major differences between the deployment of the GeoNet stack in OBU. or RSUs.

4.1.2 Software and tool requirements

Operating system: The prototype implementations of GeoNet are based on Linux Operating System (OS) with kernel 2.6.19 or higher. Kernel 2.6.19 is sufficient for pure C2CNet layer implementations. However, this limits the supported functionality dramatically, particularly IPv6 layer features. The kernel version should be 2.6.29 or higher on GeoNet OBUs where NEMO (network mobility), MCoA (Multiple Care-of Address Registration) and IPv6 multicast is required (GeoNet RSUs do not require NEMO and MCoA as there is no module “3B: Mobility Support”, however module “3C: Multicast” is required (in most cases, when GeoNet RSUs are supposed to process GeoBroadcast packets at the IPv6 layer).

Other embedded operating systems as QNX, OSEK/VDX, WinCE or similar may be used if they support IPv6 (including NEMO and MCoA extensions and MLD for multicast), and TCP/UDP.

An example for how to enable IPv6 multicast in Linux kernel 2.6.29 (or later) is provided below (MLDv2 is activated by default in newer kernel):

1. Edit the kernel configuration as follows and as indicated on Figure 5 and recompile it to support multicast:

```
Networking Support
-->Networking option
    -->IP: multicasting
    -->The IPv6 protocol
        -->IPv6: Multiple Routing Tables
        -->IPv6: source address based routing
        -->IPv6: multicast routing (EXPERIMENTAL)
        -->IPv6: PIM-SM version 2 support (EXPERIMENTAL)
```

2. Edit the following options on OBUs:

- net.ipv6.conf.all.mc_forwarding
- net.ipv6.conf.default.mc_forwarding

- net.ipv6.conf.IFx.mc_forwarding ; while IFx is

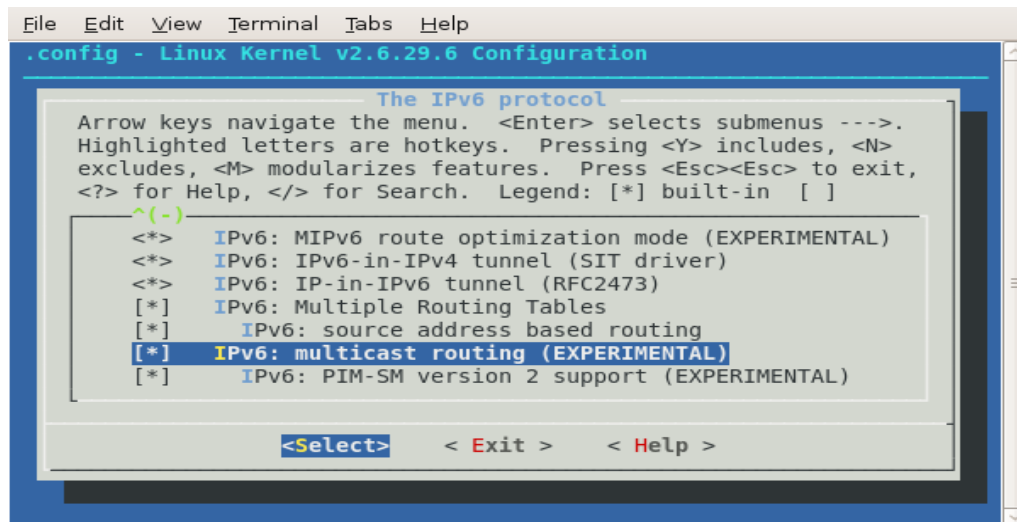


Figure 5: Linux Kernel Configuration

Multi threading capabilities are required since the effort to deploy GeoNet protocol stack in a single thread environment will be enormous.

All software modules developed by GeoNet are Linux user-space programs, and no kernel modules. The communication of different modules is based on either standard UDP or standard IPv6 sockets. UDP sockets are used in transport layer, IPv6 sockets are used in IPv6 layer. This eases the whole implementation process and does not limit the deployment of modules. However further implementations may rely on a more efficient kernel mode implementation.

Libraries and programming language: The programming language C/C++ was chosen for all software modules. No specific and only standard C/C++ libraries were used for implementation. Since C language is well standardised, highly portable and available on almost all embedded platforms, this should ease the porting and integration to 3rd party platforms. Note that 3rd party using specific libraries at the application layer must make sure about their compatibility for use in an IPv6 environment. Modules “2A: Egress Interface” and “0C: Position Sensor” are the exception as they are platform dependent. These modules must be adapted to the target platform.

Processor dependency: Please note that during integration tests GeoNet partners observed a dependency of one of the C2CNet layer implementation from processor used, in particular from vendor Intel or AMD. From this time on two binary versions are made available with different Compiler/Linker switches for C2CNet binary builds - one for Intel and one for AMD, respectively. The second prototype implementation is independent from processor vendor specifics.

Tool chain: Since GNU tool chain is of great importance for Linux and embedded systems, GeoNet relies on GNU Open Source tools. This tool chain is available for most

embedded platforms and supports porting in a perfect way. In particular following tools are used:

- GNU make: Tool for facilitation of build process;
- GNU Compiler Collection (GCC): Suite of compilers for different programming languages, supporting standardised ANSI C and C++;
- GNU Binutils: Suite of tools including linker, assembler and other tools
- GNU Debugger (GDB): Code debugging tool;
- GNU Libtool: Tool to manage component libraries.

The GNU makefiles are made available within the source code. This eases porting additionally since exact instructions are given on how to build the binaries. Cross compilation and linking statements may be easily added. Relying on the GNU tool chain, partners used different editors or IDEs (integrated development environments), depending on their companies policy and personal intentions in this area.

Tools of conformance test system: The conformance test system is based on TTCN-3 (Testing and Test Control Notation) that is standardised by several ETSI standards, <http://www.ttcn3.org/standardsuite.htm>. The TTCN-3 language is used to specify the test that runs on a dedicated test bed system. Binaries were tested in GeoNet test bed. This means that target platform and test platform must be binary compatible.

4.1.3 Development platform

Just as further information: State-of-the-art x86 systems - laptops and PCs - were used by GeoNet partners as development platforms. This should also be a good choice for embedded target platforms using the GNU cross-compiling options.

The management of source code and binary releases was carried-out by using the tool SVN on a dedicated GeoNet server. Developers collaborated using the TRAC tool that was set up on GeoNet server with ticket-based task management.

4.2 Porting and integration of GeoNet modules

4.2.1 Platform independence of GeoNet stack

Hardware platforms and operating systems handle access to hardware and software resource in very different ways. Main differentiation parameters are:

- Interrupt handling, I/O port access, event handling, call back routines;
- Kernel space / user space;
- Protect concurrent access to shared resources by semaphores or mutual exclusions (mutexes);
- Endianness - often referenced to as byte order with big-endian (most significant byte first) or little-endian (least significant byte first);
- Support for different kind of multi-tasking and / or multi-threading.

GeoNet subdivides software modules by functionality. Each software module runs in its own process, all processes are running in user space. Processes exchange information by using socket connections. Thus the concurrent access to resources is handled by operating system.

The endianness problem is solved by the use of network order library functions: *htonl* (host-to-network-long) and *htons* (host-to-network-short) functions to convert 32-bit and 16-bit values respectively from machine (host) to network order and *ntohl* and *ntohs* functions convert from network to host order.

All operations that require hardware access are moved to modules outside the core of GeoNet stack. These operations are performed by typical platform drivers. This is of course platform dependent and a matter of porting.

A characteristic example is the Position Sensor: On the hardware side the Position Sensor relies on Linux *usbserial* driver and GPS daemon *gpsd*. Once the position information is received from hardware by using the NMEA protocol, the positioning information is transformed into MNG-C2C format and provided to different GeoNet modules by socket connections. In this way the OC: Position Sensor module is used to encapsulate hardware access.

Therefore the requirements to hardware target platform integration are reduced to the minimum of three hardware / operating system dependent driver interfaces:

- Egress interface to access ETSI ITS-G5 or IEEE 802.11p compliant NICs;
- Position information interface to update position information;
- Ingress interface to communicate to IPv6 hosts attached to GeoNet OBU or GeoNet RSU (e.g. IPv6-based Application Units) – Note, nothing is to be implemented here.

For each interface there is a SAP specified thus ensuring an encapsulation of hardware / software driver dependency.

On the operating system and software library side the access is limited to standard POSIX C/C++ library routines only.

4.2.2 Management Layer

Figure 6 shows the modules of the Management layer. There are no special porting and integration requirements for Management Layer. The SAPs are specified as UDP or IPv6 port connections with following parameters:

- Socket family: AF_INET6;
- Socket type: SOCK_DGRAM;
- Address: localhost;
- Port number: Submitted as command line parameter.

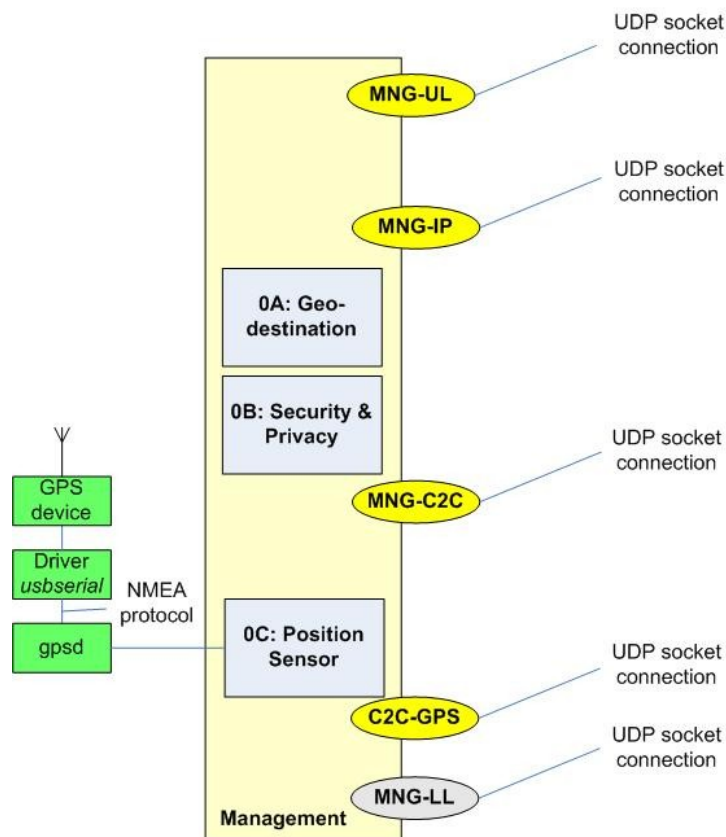


Figure 6: Management layer and related SAPs

Module 0A: Geo-destination: This module has recently been introduced in GeoNet (see [GeoNetD1.2] and [GeoNetD2.2]) but is not fully specified, nor is it implemented so far. This module is not necessary to implement and test C2CNet with a limited set of applications but will be required for wide deployment of applications in order to benefit from IPv6 GeoNetworking (note that Application Layer issues are out of scope of GeoNet). This is a standardisation issue in work items related to the Facility and Application Layer in order to support IPv6 GeoNetworking properly.

Module 0B: Security & Privacy: This module is written in C++. The source code is available and can be build with help of make tool. The implementation only uses standard C headers, no additional library are included. As for the other modules, SAP are implemented with UDP sockets. The module uses the sleep system call, that is standard since POSIX.1. Therefore the module can be compiled for almost every target platform.

Module 0C: Position Sensor: This module is required by C2CNet modules to get notified about the current position of vehicle or RSU. The interaction between C2CNet and Position Sensor is done by UDP sockets. Therefore it can be ported to any operating system providing POXIS compatibility. The C2CNet modules are completely independent of the method used for gathering position information.

Since the source code of module “0C: Position Sensor” is made available, no serious problems are expected for porting and integration. The green boxes in Figure 6 show the example of prototype implementation. The Linux GPS demon *gpsd* delivers the position information. A virtual COM port is used to communicate to GPS device. This may be different on target platform. Please check [GeoNetD2.2] for specification.

In the example implementation a class named GPSSI performs the connection to *gpsd* and realises the implementation of a position sensor. In case another more precise algorithm for positioning is used, the current implementation of Position Sensor can be serve as a basis: Only the main()-function must be changed in creating an object implementing the new position sensor. The call for creating a GPSSI-object can be removed. Further the function :

```
void PositionSensor::GPSServer::sendPostion(Position pos);
```

must be called to send new position data to C2CNet modules via the MNG-C2C SAP. The structure of position information is as specified by [GeoNetD2.2].

Note, that the position information contains accuracy information in packets submitted to MNG-C2C SAP. Therefore a more precise position sensor system will be able to inform other GeoNet modules about improved accuracy of position information.

4.2.3 Lower Layer

These modules, named as Egress Interface and Ingress Interface components, are also not core parts of GeoNet protocol stack. They are shown in Figure 7.

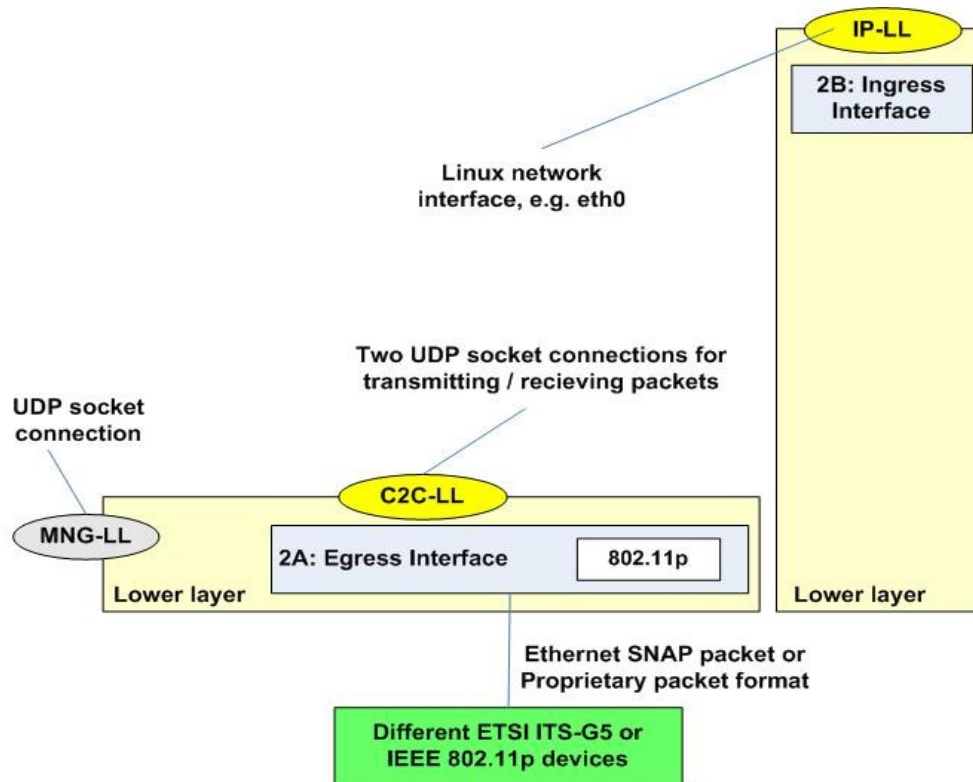


Figure 7: Lower Layer and related SAPs

In prototype implementations the module “2A: Egress Interface” is introduced to support the platform independence of GeoNet. However, in the simplest case the Egress Interface module is not needed when the target platform implements the C2C-LL SAP for communication to GeoNet C2CNet module on its own.

In most cases an Egress Interface module will be required. Most hardware vendors try to implement their drivers in such a way that they work very efficiently even neglecting layer structure or common standards. Therefore, the “2A: Egress Interface” module in most cases will be adapted to the driver interface provided by the hardware vendor of ETSI ITS-G5 or IEEE 802.11p compliant NIC on a given platform. Module “2A: Egress Interface” is implemented in C++, socket family is AF_INET, socket type SOCK_DGRAM, address port 4440 (localhost) for receiving messages from GeoNet-internal modules, and port 4441 (localhost) for sending messages to GeoNet internal modules.

The “2B: Ingress Interface” is the physical network interface link to other nodes in same subnet. Any network technology such as Ethernet IEEE802.3 or IEEE 802.11 and State-of-the-Art LLC implementations can be used. There is no module implementation necessary.

The structure of the Lower Layer architecture reflects the function of this module as interface between C2CNet modules and LLC or physical layer access. It is divided into two parts. One platform independent part interacting with the GeoNet core modules and one exchangeable part that interacts with the physical layer.

Figure 8 shows the interface between platform depend and independent code. The Lower Layer is ported to another platform by implementing the functions of this interface in C++. In that way only few classes have to be adapted for porting to a new platform.

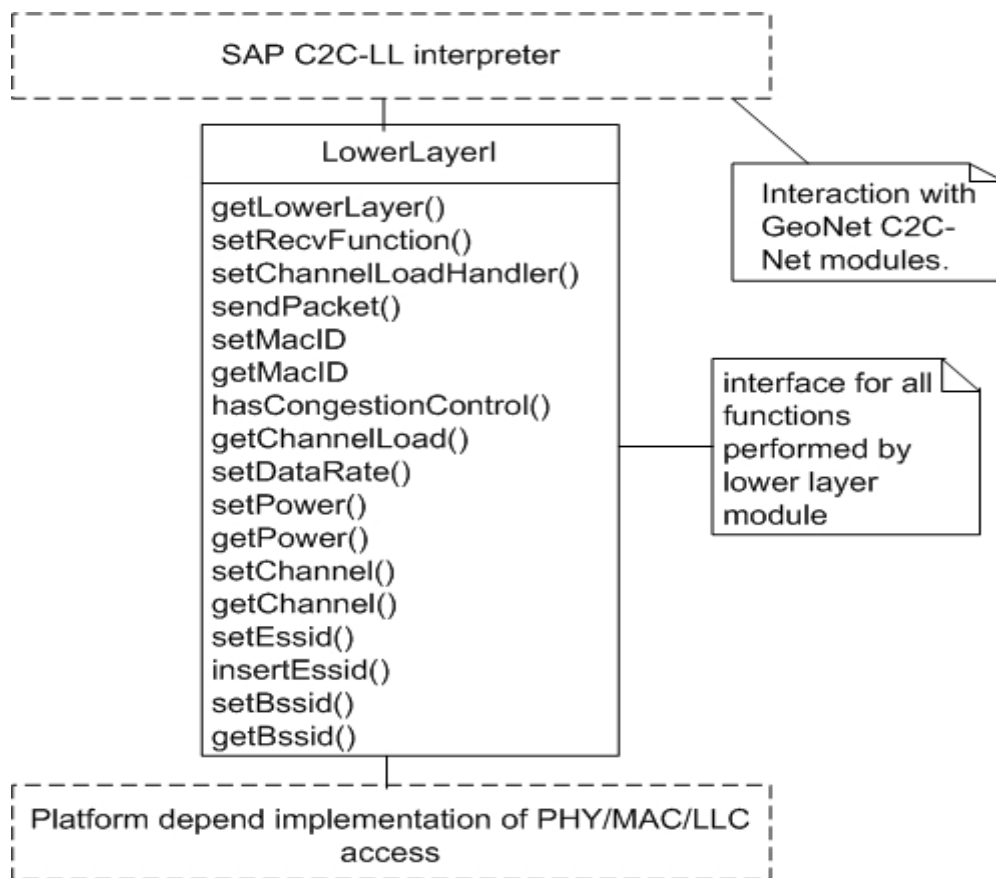


Figure 8: Interface to hardware or MAC specific functions

The first implementation of this module was done with the use of the Linux tool Wireless Extension (WE). The example of porting to CVIS platform is described later in section 28CVIS Porting example.

Note that current Linux kernel releases replace the Wireless Extension tools by more sophisticated drivers and tools of a unified WLAN stack, see <http://wireless.kernel.org/> for more details. The Wireless Extensions are deprecated. Therefore, as a future work it would be very useful to integrate drivers for ITS-G5 and IEEE802.11p into this unified WLAN stack.

4.2.4 C2CNet Layer

Figure 9 shows the structure of the C2CNet Layer of GeoNet in detail. From the porting and integration viewpoint the SAP C2C-IP is interesting. It is implemented using a tunnel interface tun. The target platform should provide a similar capability. Depending on the target system a solution must be found that enables the use of IPv6 packets at this SAP.

Using the tunctl tool in current implementation a shell script configures IPv6 and the TUN interface. An IPv6 application can be used for which the information from IPv6 to C2CNet and vice versa is sent via this TUN interface.

Details about MNG-C2C SAP are provided in Management Layer sub-clause. C2C-LL SAP is analysed in Lower Layer sub-clause.

Note that two independent reference implementations are available that follows the GeoNet C2CNet layer specification in [GeoNetD2.2]. Both of these modules are available for new target platforms on base of an individual license, see section 7.

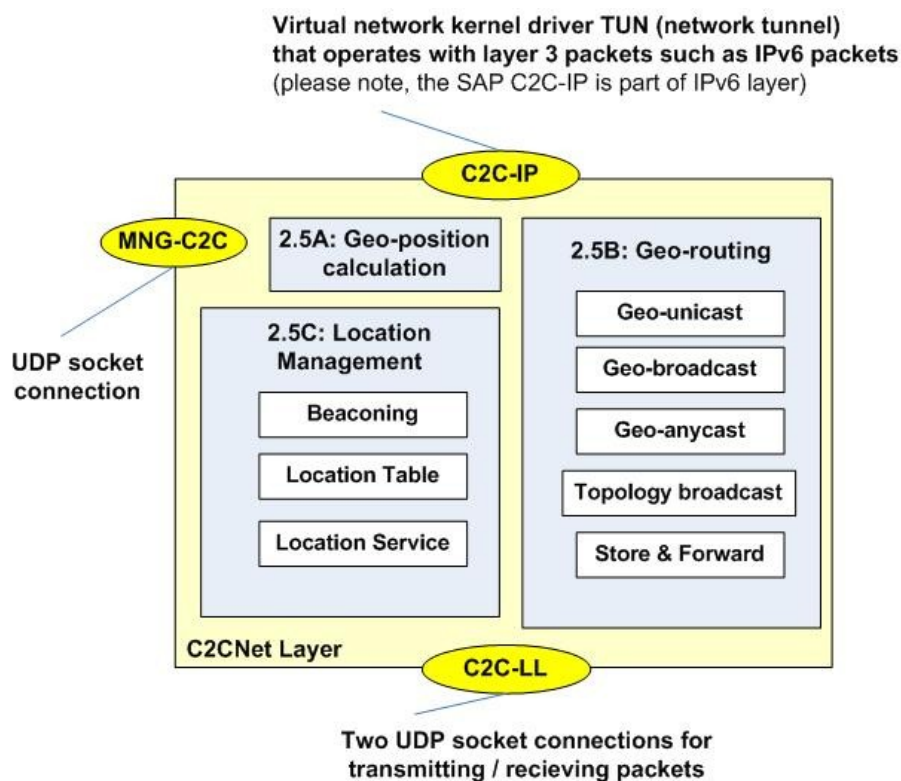


Figure 9: C2CNet Layer and related SAPs. Please note the C2C-IP SAP belongs to IPv6 layer and is shown here to ease

4.2.5 IPv6 Layer

The functions provided by Module “3A: IP Forwarding” comprises functions contained in legacy IPv6 stack and GeoNet extensions, that is sub-module “IPv6 over C2CNet” which specifies how IPv6 packets shall be transmitted over the C2CNet link. In addition, this module also includes MNPP (exchange of IPv6 prefixes of nearby vehicles in order to allow direct V2V communications and location privacy (pseudonyms) and location privacy functions. SAP C2C-IP allows to link the IPv6 and C2CNet layers (see also section on C2CNet Layer).

The functions provided by Module “3B: Mobility Support” are mostly provided by UMIP, a portion of code originally provided by the USAGI project and Nautilus6 project in Japan. UMIP itself only provides IPv6 mobility functionality, particularly NEPL (NEMO Platform for Linux) and MCoA (Multiple Care-of Address Registration). As GeoNet specific functionality, Care-of Address are configured with a network prefix (64 bits) provided by the access network (served by the GeoNet RSU) and a host suffix (64 bits) provided by the C2CNet ID (64 bits).

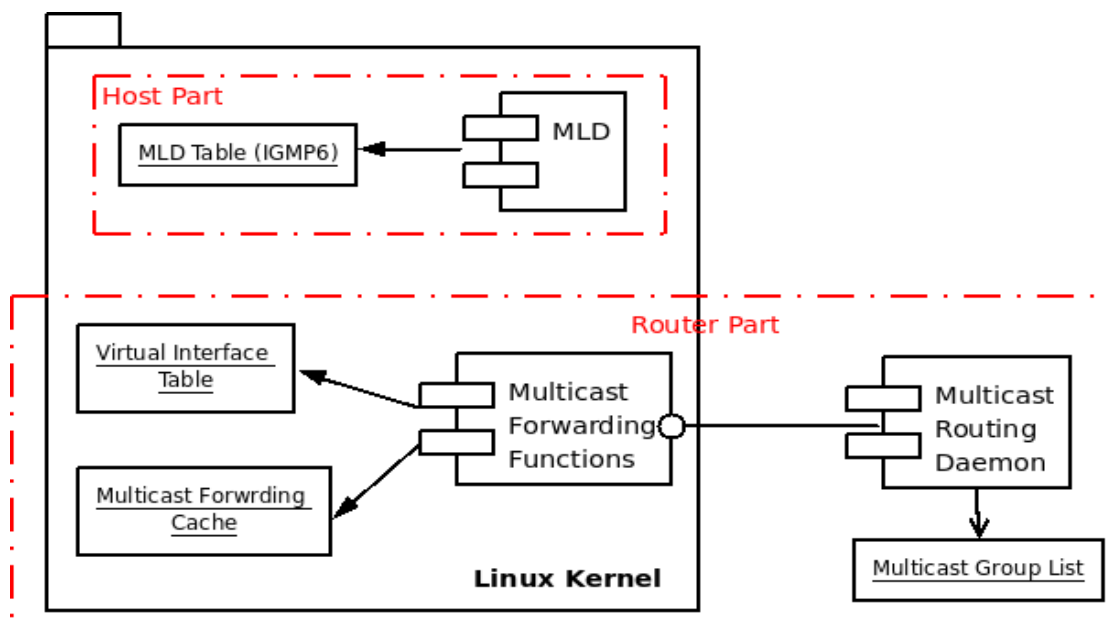


Figure 10: Application Unit running a road application and subscribed to a multicast group

The functions provided by Module “3C: Multicast” are legacy IPv6 multicast group management (MLD) and multicast routing (proxy MLD). In GeoNet implementation, a static configuration of IPv6 multicast was chosen for simplicity (see [GeoNetD2.2.] and [GeoNet7.1]). MLDv2 is natively enabled in both routers to discover the presence of Multicast listeners and hosts to report the multicast groups they are subscribed to. A router could run at the same time the router part and the listener part if it wants to receive multicast packets. In Linux systems, three tables providing information about the multicast

configuration, are administrated by the kernel and can be modified through system calls. The following files could be checked:

- `/proc/net/igmp6`: contains information about the group membership (the multicast group to which a host is subscribed)
- `/proc/net/ip6_mr_cache`: contains statistics about the sent and received packets for a multicast group and an origin address
- `/proc/net/ip6_mr_vif`: shows the multicast installed interfaces

As shown in Figure 10, the IPv6 hosts (AUs) that are running an ITS application for which packets are sent to a group of destinations in a particular GeoDestination maintain a membership table showing the multicast groups to which they are subscribed. Once the application is launched on AUs, the multicast group is added to default multicast group set on boot-up by the kernel as shown in the table above. GeoNet IPv6 OBUs and RSUs run the multicast daemon which enables multicast forwarding by updating the Virtual interface table and the multicast Forwarding Cache which are, as said before, two structures maintained by the Linux kernel. Moreover, the multicast daemon creates a list in which it saves the multicast groups on its ingress interface (connected to AUs).

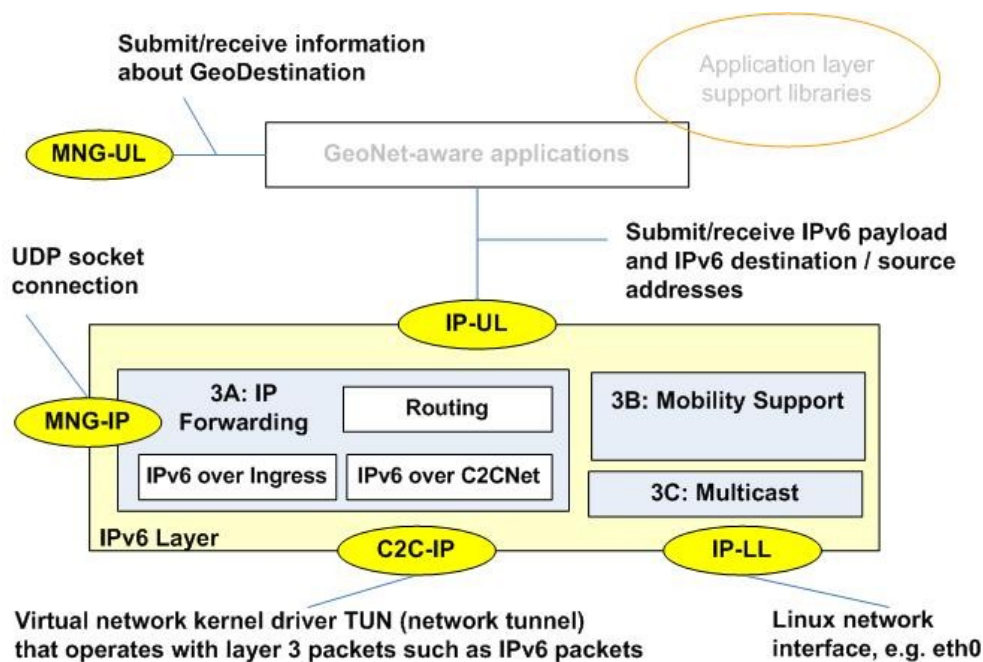


Figure 11: IPv6 Layer and related SAPs [GeoNetD3.1]

Regarding the encoding of the GeoDestination information (see SAP UL-IP and Upper Layer) into IPv6 multicast packet header, a very simple static encoding was picked up although four more flexible and scalable approaches are discussed in [GeoNetD2.2]. One approach will have to be selected as a default, but this would require more specification and possibly standardisation work. The static approach is fine at the time being for testing and using IPv6 GeoNetworking for a limited set of Geo-aware applications. This static

implementation requires a table at the C2CNet layer (in theory, this table should be implemented in the Management Layer) which binds a specific IPv6 multicast address to geographic information.

Figure 11 shows the IPv6 layer related SAPs. The SAPs C2C-IP, MNG-IP, IP-LL are discussed in previous sub-clauses. Please refer to C2CNet and Management Layer modules. IPv6 layer modules are provided in [GeoNetD6.2] software distribution and the IPv6 layer extensions implemented specifically for IPv6 GeoNetworking can be obtained separately from the GeoNet distribution.

4.2.6 Upper layers

Additionally to the information provided by IP-UL GeoNet-aware IPv6 applications need to provide information about the geographic destination area (GeoDestination) for lower layers (IPv6 and C2CNet) to route the packets to the appropriate geographic area. This information will be received by the application and ITS facility layers via SAP MNG-UL. However, the ITS application and facility layers are out of scope of GeoNet. Also transport layer is not included in GeoNet specifications. Instead the default mechanisms of TCP/UDP are used. However, as mentioned in sub-clause Management Layer at this level there is more standardisation and specification work needed in the future in order to ease the use of IPv6 GeoNetworking.

4.2.7 Tools for testing

The tools for testing are described in [GeoNetD4.1], section 3.4.3, see <https://geonet-project.eu/geonet/browser/Documents/WP4/Deliverables/D.4.1/GeoNet-D.4.1.odt>. Since binary software modules are tested, the TTCN-3 test suite should be able to run the binary code of the embedded target device. Also an emulation environment could be used on the test host. The test-system of GeoNet makes additional use of the following tools:

- picottcn: compiler of TTCN-3-language ,
- T3DevKit: toolkit to ease development of test suites ,
- eclipse: editor for TTCN-3-files ,
- boost C++: C++ Libraries for many different platforms ,
- PHP: script language used to process the tests .

4.3 GeoNet Installation and User Guide

Since the GeoNet Handbook [GeoNetD8.3] already provides a detailed installation and user guide for the GeoNet stack this will not be repeated here in this document. Instead the readers attention is directed to section 6 in [GeoNetD8.3].

5. CVIS Porting example

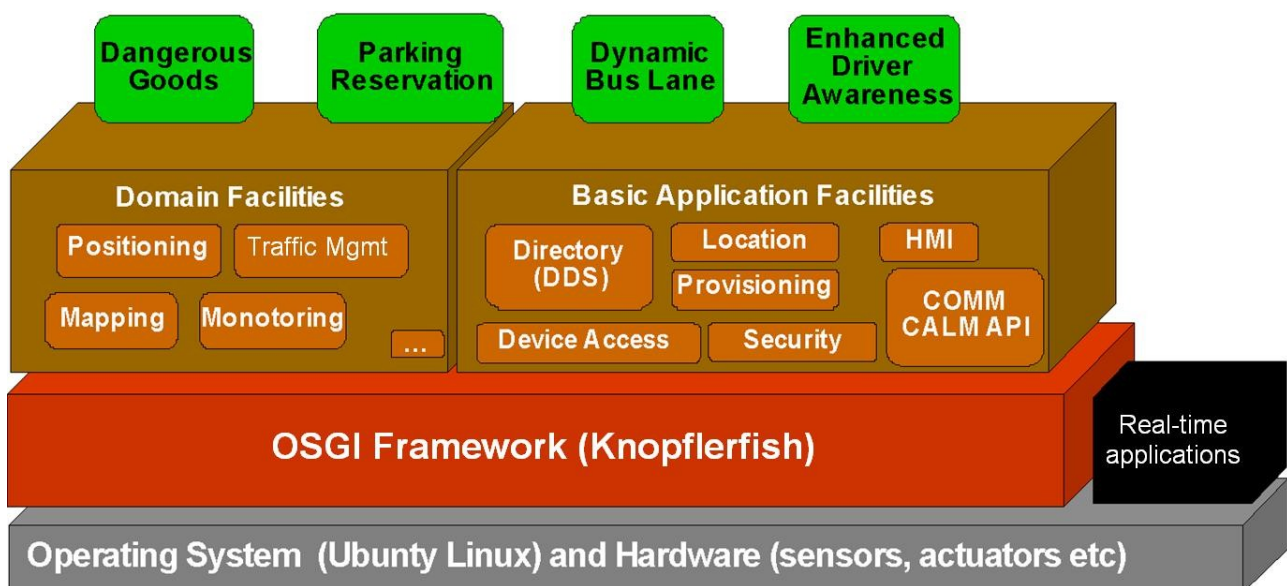
This section delivers a guideline and the experience of porting and integration to CVIS platform. Two options are discussed: A lightweight and a full integration of GeoNet into CVIS.

The lightweight integration is based on CVIS OSGi architecture. The GeoNet OBU (Mobile Router) runs on a separated controller that is connected by Ethernet to a CVIS Host. This integration will be detailed in paragraph 5.1.

The second approach performs a full integration into CVIS architecture. It will be explained in paragraph 5.2. The third paragraph describes how the porting to CVIS was realised right now. At the end of this chapter an outlook is given, how to improve this integration in future steps.

5.1 Lightweight integration

The goal of lightweight type of integration is to enable the use of GeoNet protocol stack for CVIS applications in a very easy way. Figure 12 shows the CVIS software stack:



Source: Vilmos Nebhaj, Ramsys, Presentation „CVIS Software Environment“ held at GeoNet Meeting July 8th in Frankfurt (Oder)

Figure 12: : CVIS Software stack.

Figure 12 shows that almost all CVIS applications rely on OSGi Framework Knopflerfish. Therefore developing an OSGi module compliant with IPv6 GeoNetworking will make CVIS applications GeoNet aware. In order to exploit IPv6 GeoNetworking, CVIS application must become Geo-aware and be able to transmit packet using IPv6 multicast. This is possible using the right function call using the same AF_INET6 socket as used

currently within CVIS. This requires a few lines of code as long as the static GeoDestination to IPv6 multicast encoding approach as described in section 4.2.5 is used and as long as IPv6 multicast addresses are configured statically as implemented in GeoNet, see section 4.2.4 . Learning IPv6 multicast addresses dynamically would require the use of MNG-IP SAP and module “0A: Geo-destination”, which is not yet specified.

Figure 13 provides the architecture of lightweight integration that was created in direct cooperation between CVIS and GeoNet project partners. On the CVIS Box (Host) a GeoNet OSGi module provides an IPv6 based connection to the GeoNet OBU (Mobile Router / MR) is implemented. This could be either a Vehicle ITS Station or a Roadside ITS Station as explained in [GeoNetD1.2]. The GeoNet OBU is able to communicate to all other GeoNet OBU and GeoNet RSU in the GeoNet domain i.e. through C2CNet.

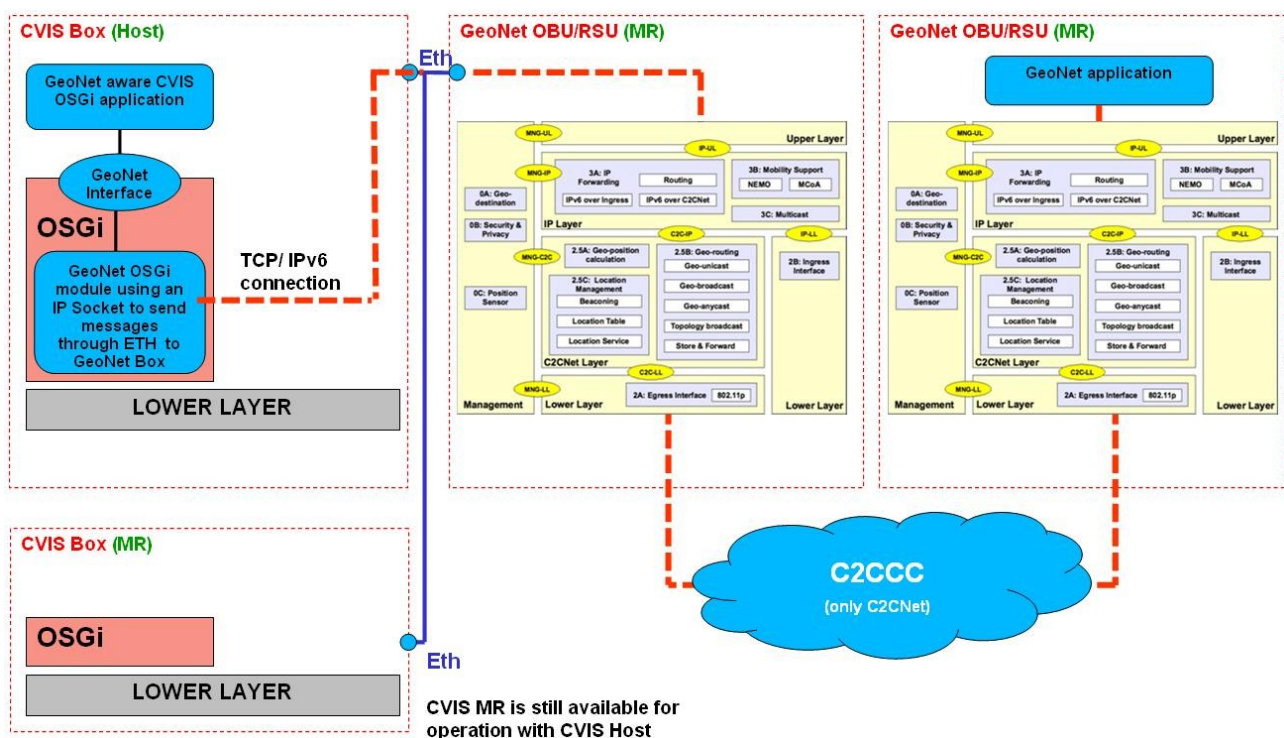


Figure 13: Architecture of lightweight integration of GeoNet into CVIS platform

The structure of CVIS and GeoNet software remains unchanged.

The advantage of this approach is the simplicity of integration. There is no need to transfer any IPR or source code between CVIS and GeoNet partners. The disadvantage is the need for separate GeoNet Box (MR) what may be accepted only for research purposes.

This integration example was developed within GeoNet's WP6. However, within GeoNet project runtime there couldn't be achieved an agreement with CVIS project management to prove this implementation e.g. by an Enhanced Driver Awareness (EDA) application as suggested by GeoNet project management several times.

The source code of this OSGi implementation example is freely available within D6.2 software deliverable, see [GeoNetD6.2].

As a future work this GeoNet OSGi module could be enhanced in such a way that it exposes both the IP-UL and the MNG-UL SAPs.

5.2 Full GeoNet integration

In full integration GeoNet will run directly on the CVIS MR Box as an additional software module and provide all GeoRouting functionality to CVIS applications. CVIS applications may use IPv6 GeoNetworking in the same way as any other protocol supported by CVIS. GeoNet protocol stack will access IEEE802.11p or ETSI ITS-G5 compatible hardware without disturbing the parallel use of these interfaces by other protocols.

5.2.1 Modules to port

Porting of Management Layer: In the Management Layer the Position Sensor module is of interest for porting. But since GeoNet focus on the routing algorithm no adaptation of the Position Sensor is done. The gpsd based prototype implementation is used instead. The module Security & Privacy can be compiled on CVIS platform without further actions.

Porting of Lower Layer module: Module “2B: Ingress Interface” (see Figure 4) must not be ported, since CVIS is also running on Linux, which supports the access of applications directly above LLC. Only the module “2A: Egress Interface” must be changed to make use of special CVIS-driver implementation.

The CVIS network layer is implemented in the kernel. The modules of GeoNet are implemented in user space. As a consequence the structure of GeoNet Lower Layer must be adopted to CVIS driver model for porting. To access the functions of CVIS kernel implementation, the Lower Layer module separated into two parts.

The first part runs similar to the current Lower Layer module in user space. It will forward the messages of GeoNets C2CNet module to the second part of the Lower Layer. The second part is a kernel module. This kernel module provides an own socket-type for receiving packets from Lower Layer. It creates an GeoNet stack specific Virtual Communication Interface (VCI) for GeoNet usage.

To interact with the communication interfaces (CI) of a CVIS platform the CALM device driver framework is used. It enables access to the communication interfaces, the virtual communication interfaces and the driver of the ETSI-ITS-G5 hardware.

To achieve a full integration of GeoNet the CVIS software has to be extended. GeoNet packets are recognised by Ethertype 0x0707. For the CALM M5-driver the GeoNet packets should have a special meaning and the driver shall handle the parameters in

GeoNet packets. This applies mainly to the packet priority. Since GeoNet had no access to source code this could not be approved.

Porting of C2CNet Layer: The CVIS platform is compliant to the requirements listed in D3.1, therefore the two available GeoNet implementations of the C2CNet layer are both able to run on it. Within WP6, it was managed to run both NEC and Hitachi daemons on CVIS platform without any problem.

Porting of IPv6 Layer: Since CVIS is also IPv6 based, the applications can use IPv6 in the same way as GeoNet does in its prototype implementations. Nothing must be changed here right now. There is no porting and implementation required.

Porting of Test tools: The tools (see 4.2.7) used for testing are public available and can be compiled for this platform. The conformance testing tools produced within WP4 are platform independent, and thereby support porting of project results.

5.3 Implementation on CVIS platform

There are two partly implementations of the introduced porting methods. The paragraph 5.3.1 shows the implementation of the lightweight integration. Paragraph 5.3.2 illustrates the first step of full integration implementation done in the GeoNet project. Both implementations are unilateral at this moment, since they are done from GeoNet side.

5.3.1 Lightweight GeoNet integration to support OSGi GeoNet aware applications

It is assumed for this integration that GeoNet runs on the mobile router or a RSU and not on the host system. To make a CVIS-application GeoNet aware it must be able to:

- interact with GeoNet management modules (Geo-destination, PositionSensor and Security & Privacy), which run on the mobile router
- sending and receiving of messages sent thorough GeoNet

To access the GeoNet stack on the mobile router a OSGi-module called *GeoNetForward* was created. This module is running on CALM-host and works like an interface for GeoNet on CVIS platforms. GeoNet aware applications will utilize this *GeoNetForward* module to use the GeoNet functionality.

GeoNetForward module works as follows: All actions will be performed by sending a broadcast to the subnet of mobile routers to a defined port using IPv6-UDP packets. On the mobile router another application must be active to receive UDP packets. This 'forward' application on the mobile router (or RSU) works as a proxy. All actions requested by the OSGi modules will be forwarded by this application.

Right now there is a type of 'Hello world' implementation available. The OSGi module *GeoNetForward* is created, but the interfaces to the OSGi-applications is not specified.

5.3.2 Full integration

Only the Lower Layer module is adopted, since the other module can work on this platform without change.

The implementation done right now is a user space version of the full integration. The Lower Layer interacts with the CALM device driver framework in kernel space over netlink sockets. For integration the CVIS library *libcalmmmedia* is used. This library contains functions and headers needed to access the CALM device driver framework. In that way it is possible to access and configure the required (virtual) communications interfaces for use by GeoNet.

The Figure 14 shows the Lower Layer module and it changes for porting and integration. For sake of clarity only classes and functions are shown that are involved in porting.

The communication between Egress Interface and LLC sub-layer is based on raw sockets. The implementers are aware, that raw sockets have been known to have problems with portability what could cause problems when using other platforms. However, raw sockets are easy to handle, not Linux specific and very effective to exchange information between different components. Therefore, it should be easy to find workarounds for possible porting problems.

Functions of class *CalmLowerLayer* (Figure 14): The Class *CalmLowerLayer* provides the implementation of the Lower Layer in CVIS. However, it does not configure the network device. This is done by the implementation of device type *CalmDevice*. Therefore, setting of channel, *essid* or *bssid* by *CalmLowerLayer* will take no effect because the design assumes an ad-hoc network.

Functions of class *CalmDevice* (Figure 14): The class *CalmDevice* handles the VCI's associated by the base device name. It is able to configure the VCI's with help of the functions in the library *libcalmmmedia*. Further it sends and receives messages. All parameters to transmit a message (TX-power, channel, priority...) will be set before sending. Since there is no direct support of GeoNet headers in the device driver of CVIS all preferences of the M5 device will be repeated for each packet sent out.

Because of the Lower Layer runs as a normal user space program other applications are able to change the settings before the message is transmitted by the radio. A GeoNet aware M5 device driver could avoid such behaviour.

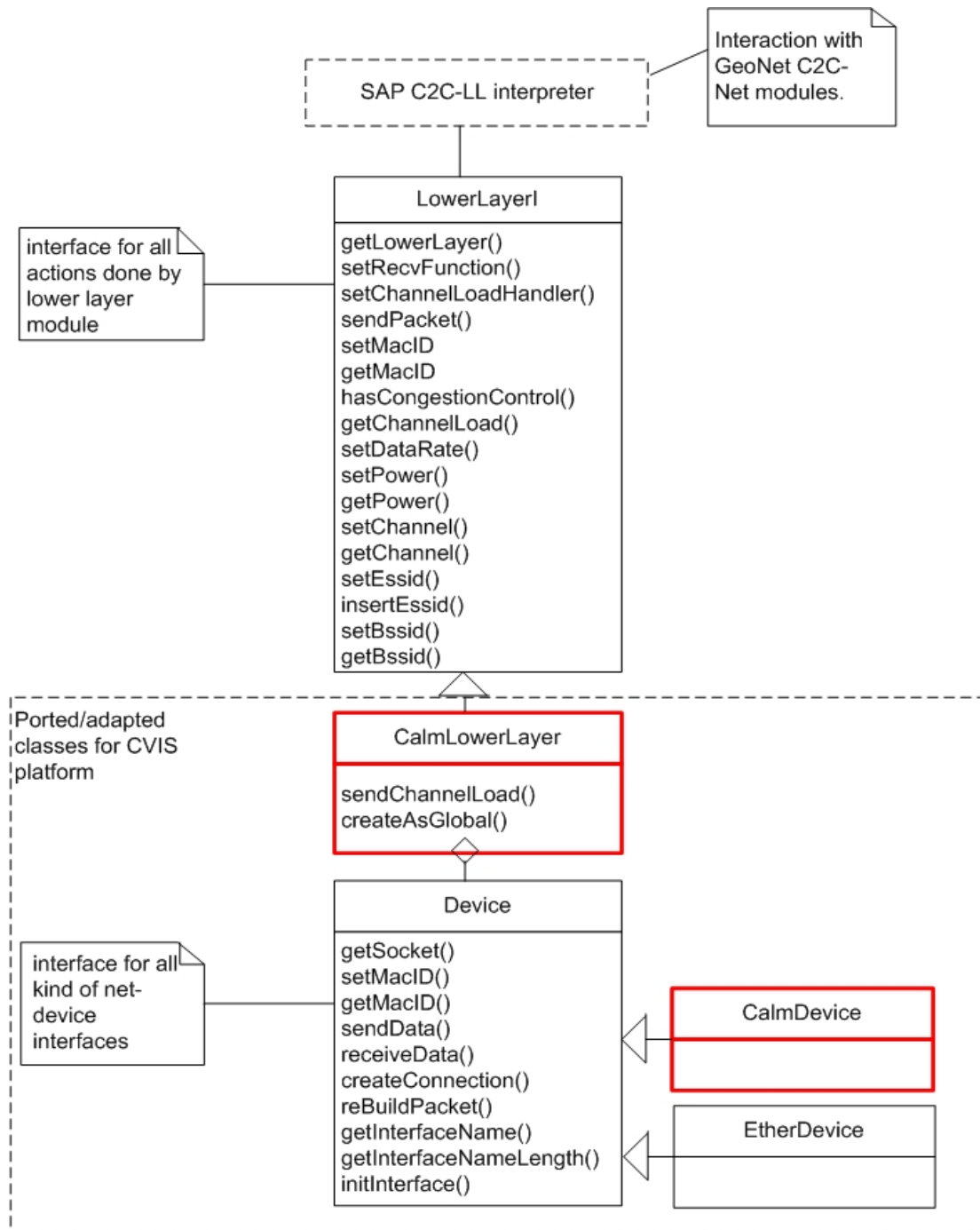


Figure 14: Lower Layer classes involved in porting

5.3.3 Evaluation of full integration

The disadvantage of current implementation is that not all benefits of a full integration are available. In current implementation no special VCI for GeoNet is created, instead the available default CVIS VCIs are used. This causes performance losses. Further there is no kernel integration of GeoNet from CVIS site. Therefore the M5 driver of CVIS will not handle the parameters in a GeoNet packet. This integration step lies outside of the scope of GeoNet and can only be performed with support of CVIS project. The Lower Layer of GeoNet simulates this behaviour by changing the settings of the VCI. Every time when a new packet is sent the settings of VCI are updated.

Unfortunately there was no porting and integration support of CVIS project available. Under these circumstances the advantage of this kind of integration is, that no kernel modules must be developed from GeoNet side. For a first integration step this is the functionality needed to show that GeoNet runs on CVIS platform. With this integration GeoNet gains access to the wireless medium – the M5 devices - to work. The proper functioning of GeoNet routing mechanisms are not restricted by this unilateral porting.

5.4 Porting and integration results - outlook

It was shown, how to port GeoNet to CVIS platform in a first preliminary implementation. The first step of integrating GeoNet to CVIS platform was done. The complete functionality of GeoNet is already available with the current implementation. However, there are further enhancements necessary to make CVIS applications and M5 drivers GeoNet aware.

As mentioned before this is an unilateral porting by GeoNet. The adaptation of CVIS M5 driver will probably improve performance. In this case GeoNet packet's priority and transmit power settings will be used by the driver on a per-packet base to increase the performance of GeoNet on CVIS platform.

There are further possibilities to improve the GeoNet implementation. In current implementation the GeoNet stack must always scan all VCIs to find a new peer connection. There is no acknowledgement or event available, when a new VCIs is created. However, this is possible with help of the CALM device driver framework, when GeoNet Lower Layer runs as a kernel module. This is only one example how to increase the performance.

6. Conformance testing

In order to ensure the correctness of porting and integration, a protocol and profile conformance testing should be done to determine whether the target system meets the GeoNet functionality as specified in [GeoNetD2.2].

For conformance testing purpose GeoNet partner BroadBit developed in work package WP4 a test bed and test procedures that follow the standardised TTCN-3 (Testing and Test Control Notation version 3) test methodology and tools. Test bed, scripts and test procedures are part of deliverable [GeoNetD4.1] and will be made available for conformance testing of future GeoNet implementations by BroadBit.

Since all the details are well described in [GeoNetD4.1] this chapter provide only a short overview to highlight the specifics from porting and integration point of view.

6.1 Benefit of compliance tests

A common conformance test plan is available and may be used by 3rd parties. Many test are carried-out. Conformance testing is a real help to implementers since problems and errors may be detected in a very early stadium even before performing component tests. It is well known that the earlier a software error is detected the lower are the costs of eliminating it.

Conformance testing developed within GeoNet project is a powerful tool that may reduce development time and easy debugging. It will increase interoperability between implementations of different providers.

Since the IUT (Implementation Under Test) runs on a test platform that is different to the target platform, this conformance test is also a test of platforms independence.

6.2 Conformance test basic understandings

The test purpose (TP) is a non-formal high-level description of a test, mainly using a wiki text. This description is the basis for the formal test specification. The formal GeoNet test specification is developed with help of TPs and the TTCN tool “Abstract Test Suite”. The existing test specifications may be helpful for 3rd parties.

Test object is an IUT (Implementation Under Test). Valid behaviour and/or invalid behaviour test cases are carried out. For both kinds of test cases the positive outcome (compliance) is proved by comparing expected behaviour to real behaviour of IUT:

- Valid Behaviour (BV) tests: Verifies that the IUT reacts in conformity with the specified functionality after receipt of *valid* input information.

•Invalid Behaviour (BI) tests: Verifies that the IUT reacts in conformity with the specified functionality after receipt of a *syntactically, semantically or functionality invalid* event.

6.3 Test environment and test step sequence

Figure 15 provides an overview of GeoNet Conformance testing. Implementations are tested against specification in the BroadBit conformance test bed. IUT are tested for valid (BV) and invalid behaviour (BI). The tests are carried out for binaries in a well defined environment with fixed sequence of steps. Both the module functionality and the conformance to SAP definitions can be tested by injecting, receiving and parsing packets. Reports and log files are created.

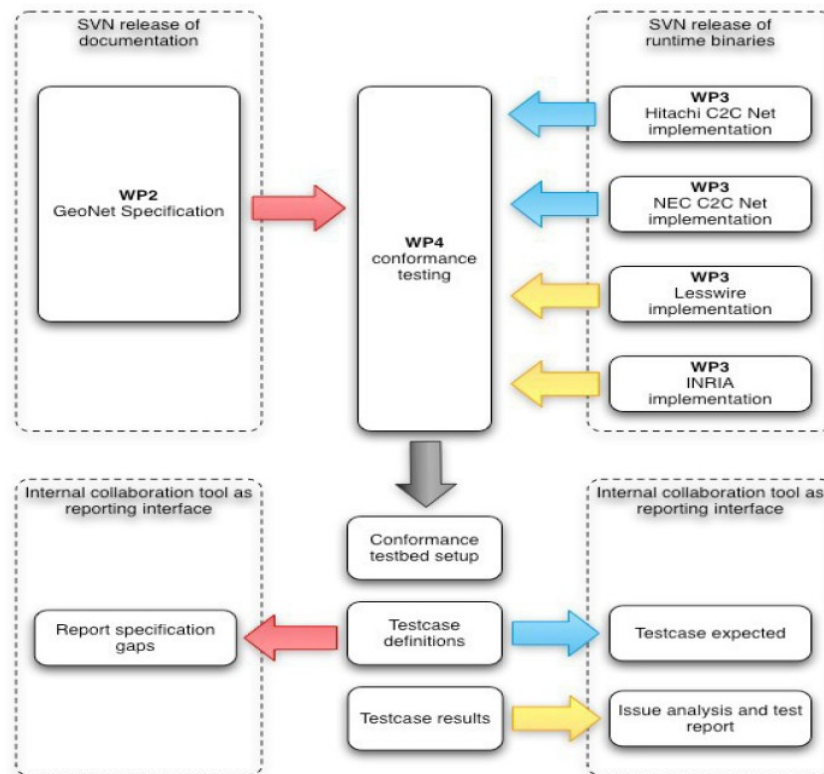


Figure 15: Overview of GeoNet conformance testing []

The environment and test sequence are available for the following modules and SAPs:

- C2CNet implementations of Hitachi and NEC
- Position Sensor and Lower Layer implementation of lesswire
- IPv6 module implementation of INRIA.
- The runtime environment consists of:

- Operating system Linux, kernel: 2.6.28-15-generic
- Wireless device card with “Atheros” chip set
- GPS device simulator

The tests are carried-out by BTT (BroadBit Test Tool) as a comprehensive TTCN tool for conformance and protocol testing. BTT supports both Binary and ASN.1 modules.

6.4 Test results

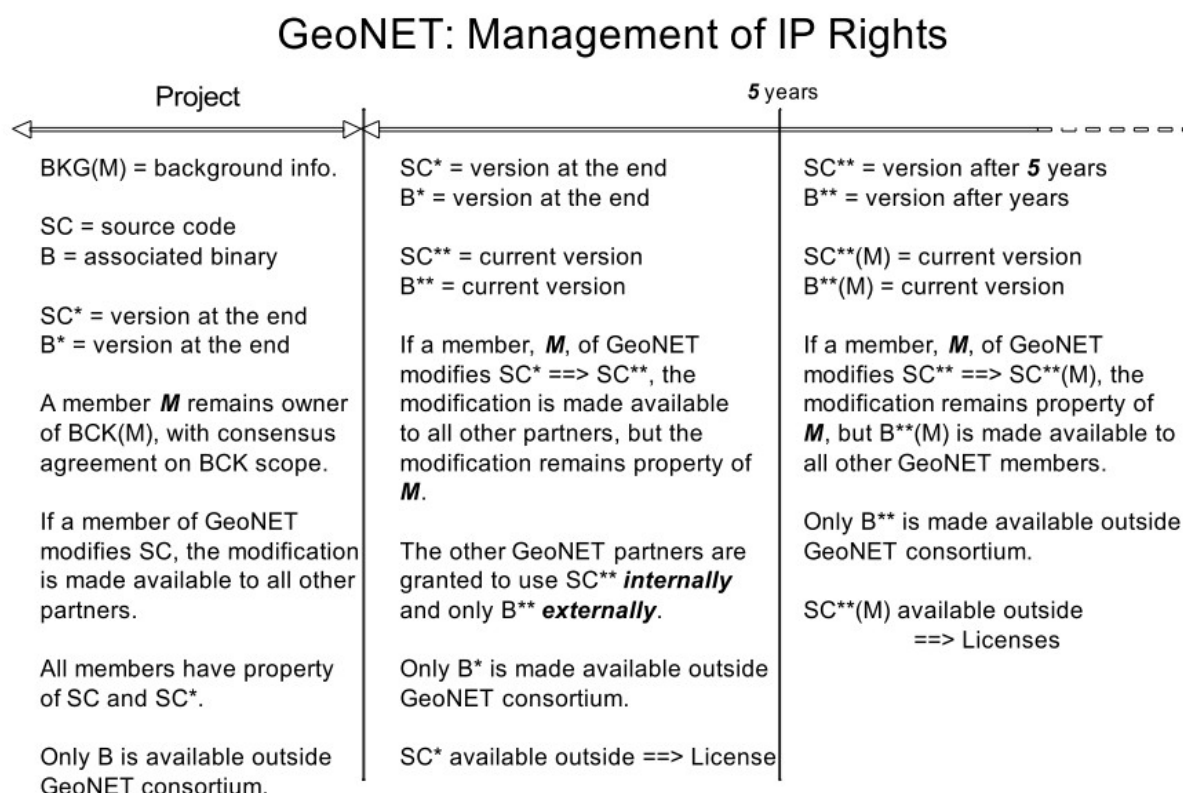
Test results are documented by reports and log files. In case of unexpected behaviour the test case failed and a TRAC ticket is created.

Thus the conformance to [GeoNetD2.2] specification can be proved and documented.

7. License policy and porting support

7.1 Management of IPR and license policy

Intellectual Property Rights (IPR) are regulated by GeoNet consortium agreement and depicted in Description of Work (DoW) in sub-clause “B.3.2.3 Management of Intellectual Property”, see [GeoNetDoW]. Figure 16 is copied from DoW and provides an overview:



Where:

- ‘use’ means incorporation into prototypes and products
- ‘internally’ means within GeoNet partners
- ‘externally’ means with respect to some party other than GeoNet partners
- ‘property’ means ownership right over a certain software code.

Figure 16: GeoNet management of IPR [GeoNetDoW], chart on p.62

According to these IPR, only binary code is made available outside the consortium, except for cases where license agreements are closed after the end of project. At the end of project all GeoNet members have the ownership right of their source code they

developed. Within 5 years after project's end all source code is made available to all partners within the GeoNet project for internal use.

It is explicitly stated, that both the interface and the functional specifications created within GeoNet project are published, with the objective of promoting their standardisation, see [GeoNetD1.2] and [GeoNetD2.2]. All binary software and major parts of source code created within GeoNet project will be made available to external parties on fair and non-discriminatory basis.

Figure 17 provides an overview of partner contribution. and the related IPR model for GeoNet protocol stack, see also [GeoNetD3.1], sub-clause “2.1.2 Partner contributions” (Partners in alphabetic order):

- **BroadBit:** Developed SAP TTCN-3 implementation and test routines for conformance testing. Maintained the SVN / TRAC system and developer SDK environment.

License model: Provide software and support on request and on base of individual license agreement.

- **Hitachi:** Developed independent C2CNet software module, utility software and corresponding documentation.

License model: Provide binary code on request and on base of individual license agreement.

- **IMDEA:** Integrated IPv6 related functionalities on security and mobility.

License model: Provide binary code on request and on base of individual license agreement

- **INRIA:** Implemented advanced IPv6 features (particularly IPv6 over C2CNet) and integrated the code with IPv6 specific functionalities (IPv6 mobility, IPv6 multicast) and

License mode: Provide source code on request according to GPL2 (<http://www.gnu.org/licenses/gpl-2.0.txt>)

- **lesswire:** Implemented Egress interface module with SAP C2C-LL prototype and position sensor software.

License mode: Provide source code on request according to GPL2 (<http://www.gnu.org/licenses/gpl-2.0.txt>)

- **NEC:** Developed independent C2CNet software module, utility software and corresponding documentation.

License model: Provide binary code on request and on base of individual license agreement.

Note: All GeoNet modules are Linux user space implementations and therefore not affected by Linux kernel copyright and IPR restrictions.

The GeoNet partners Hitachi and NEC brought into the GeoNet project a strong background and expertise of earlier developments in the area of GeoNetworking. These partners developed two independent prototype implementations. This is useful for standardisation, but also for exploitation since their characteristics can be compared and a quality competition of implementations is initiated. In turn, these implementations are not public available on source code level. However, these partners are willing to provide binaries, as regulated by consortium agreement, to 3rd parties on fair and non-discriminatory basis.

7.2 Getting support

GeoNet partners are highly motivated to support interested 3rd parties to use GeoNet protocol stack in their own ITS applications and therefore, to provide help for porting and integration.

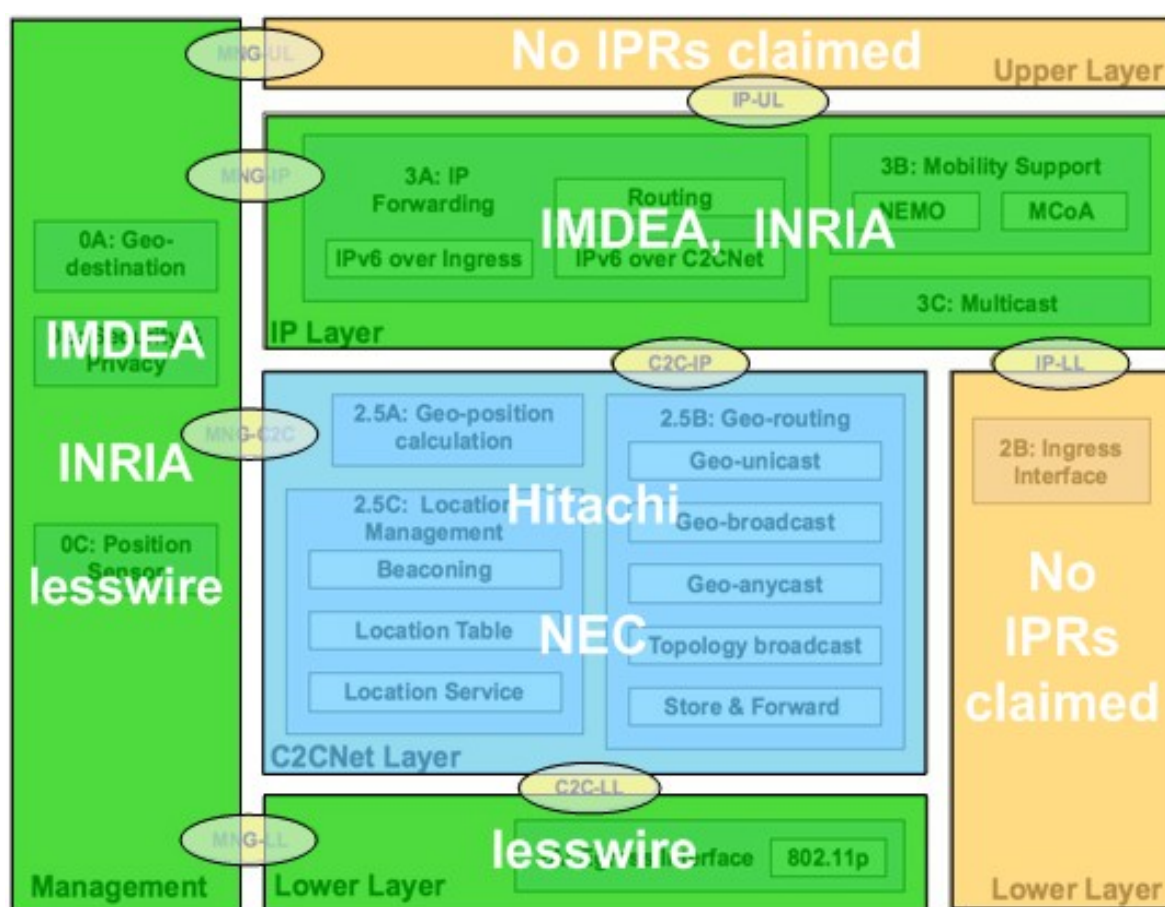
Support is provided by GeoNet partners for the modules as indicated in Table 2.

GeoNet partner	Porting and integration support for following modules
BroadBit	TTCL-3 scripts and conformance testing for modules developed by interested 3 rd parties; Testing compliance to GeoNet specifications according to [GeoNetD2.2].
Hitachi	C2CNet modules implementation and related utilities.
IMDEA	Location privacy module.
INRIA	IPv6 layer modules implementation (particularly IPv6 over C2CNet) and configuration (mobility, multicast) and C2C-IP SAP
lesswire	Position sensor and egress interface; Support for integration of ETSI ITS-G5 and IEEE802.11p compliant hardware devices

NEC	C2CNet modules implementation and utilities
-----	---

Table 2: Mapping of GeoNet SAPs to SAPs of European ITS architecture

Annex C: Getting support for porting and integration delivers the addresses of GeoNet partners for porting and integration support.



Note: Mainly Geonet partner BroadBit developed the TTCL-3 implementation of Service Access Points (SAPs)

License model

- Source code available according to individual license offered by Geonet partner
- Binaries available according to individual license offered by Geonet partner
- No IPRs claimed

Figure 17: Contribution of GeoNet partners

Annex A: Terminology & Acronyms

Acronym	Meaning
MR	Mobile Router – the GeoNet OBU. The term MR is used for simplicity and easier understanding.
AR	Access Router – the GeoNet RSU. The term AR is used for the simplicity and easier understanding.
TTCN-3	Testing and Test Control Notation 3
IETF	Internet Engineering Task Force
ETSI	European Telecommunications Standards Institute
ISO	International Organization for Standardization
CVIS	Cooperative Vehicle Infrastructure Systems
MAC	Media Access Control
LLC	Logical Link Control
NEMO	Network Mobility Support in IPv6
CALM	Communications Access for Land Mobiles
ETSI ITS-G5	ETSI Intelligent Transport Systems- G5
OSI	Open Systems Interconnection Reference Model
OBU	On Board Unit
ID	Identification
WP6	Work Package 6

SAP	Service Access Point
GPS	Global Positioning System
VCI	Virtual Communication Interface
CI	Communication Interface as defined by
M5	Wireless medium using frequencies in 5.9Ghz ITS frequency band as defined by [ISO-21215]
OSGi	Java-based service platform that can be managed remotely; Specified by OSGi Alliance, see, http://www.osgi.org .

Annex B: Contributors

The following people have contributed to this GeoNet document

- Thierry Ernst - INRIA
- Wilfried Lohmann - Lesswire
- Carsten Schulze - Lesswire
- Maria Goleva - NEC
- Marco Gramaglia - IMDEA Networks
- Hamid Menouar - HITACHI
- Ines Ben Jemaa - INRIA
- Manabu Tsukada – INRIA
- Lajos Nagy - BroadBit

Contributions from other than GeoNet project partners in clause 3.1:

- András Edelmayer - Ramsys
- Hans-Joachim Fischer - ESF
- Attila Simon - Ramsys

Annex C: Getting support for porting and integration

GeoNet partner	Address
BroadBit	BroadBit Kft., 1028, Kolozsvár ut. 12, Budapest, Hungary Tel. +36 30 218 9659 / +36 30 218 9651 andras.kovacs@broadbit.com lajos.nagy@broadbit.com
IMDEA	Fundación IMDEA Networks, Avda. del Mar Mediterraneo, 22, 28918 Leganes, Madrid, Spain Tel. +34 91 481 62 10 (IMDEA secretariat) Contact person: Marco Gramaglia (marco.gramaglia@imdea.org)
INRIA	INRIA – Project-Team IMARA - Domaine de Voluceau - Rocquencourt - B.P. 105, 78153 Le Chesnay Cedex - France Tel. +33 1 3963 5930 (Thierry Ernst, GeoNet Technical Coordinator) thierry.ernst@inria.fr http://www.lara.prd.fr/
Hitachi	Les EcoLucioles, 955 route des Lucioles, 06560 VALBONNE, France Tel. +33 4 8987 4100
lesswire	lesswire AG; Im Technologiepark 1, D-15230 Frankfurt (Oder), Germany, Phone +49 335 5656 926 Albert-Einstein-Straße 16, 12489 Berlin, Germany Tel. +49 30 6392-8283 info@lesswire.com , http://www.lesswire.com
NEC	NEC Laboratories Europe, Network Division, Kurfürsten-Anlage 36, 69115 Heidelberg, Germany Tel. +49 6221 4342 0

Annex D: References

[ETSI-ES-202-663] ETSI “Intelligent Transport Systems; European profile standard on the physical and medium access layer of 5 GHz ITS” ETSI ES 202 663 V1.1.0 (2010-01), January 2010

[ETSI-TS-102-665] ETSI. “Intelligent Transport Systems (ITS); Vehicular Communications; Architecture”. ETSI, Work in Progress, December 2009.

[GeoNetD1.2] GeoNet. “Final GeoNet Architecture Design”. GeoNet Deliverable D1.2, January 2010.

[GeoNetD2.2] GeoNet. “Final GeoNet Specification”. GeoNet Deliverable D2.2, January 2010.

[GeoNetD3.1] GeoNet. “GeoNet Development Results”. GeoNet Deliverable D3.1, January 2010. Confidential.

[GeoNetD4.1] GeoNet. “GeoNet Conformance Test Plan and Results”. GeoNet Deliverable D4.1, January 2010.

[GeoNetD5.1] GeoNet. “GeoNet Emulation Environment Results” GeoNet Deliverable D5.1, February 2010.

[GeoNetD7.1] GeoNet. “GeoNet Experimentation Results” GeoNet Deliverable D7.1, February 2010.

[GeoNetD8.2] GeoNet. “GeoNet Dissemination and Use Plan” GeoNet Deliverable D8.2, February 2010.

[GeoNetD8.3] GeoNet. “GeoNet Handbook” GeoNet Deliverable D8.3, February 2010.

[GeoNetDoW] GeoNet. Grant agreement no.: 216269. Grant agreement for small or medium-scale focused research project - STREP - CP-FP-INFISO. Annex I - “Description of Work”, Date of approval of Annex I by European Commission: November 10, 2007.

[GeoNet-Integration] A. Edelmayer, H-J. Fischer, W. Lohmann, A. Simon “Meeting Minutes – How to integrate GeoNet in ITS architecture”. Document version 1.1, 21 January 2010. GeoNet internal document.

[IEEE802.11p] IEEE “Draft Standard for Information Technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications; Amendment 7: Wireless Access in Vehicular Environments”. IEEE P802.11pTM/D9.0

[ISO-21215] ISO DIS 21215:2009, "Intelligent Transport Systems – Communications access for land mobiles (CALM) – M5". ISO FDIS specification 21215, ISO, October 2009. Work in progress.

[ISO-21217] ISO TC204 WG16. "Intelligent Transport Systems – Communication access for land mobiles (CALM) – Architecture". ISO FDIS specification 21217, ISO, October 2009. Work in progress.