Politecnico di Torino
Anno Accademico 2018/2019

Tesi di Laurea Magistrale

Corso di Laurea Magistrale Ingegneria Informatica

Analisi, Progettazione e Sviluppo di Cellular-V2X Software Utilities

Relatore
Prof. Gianpiero Cabodi
Supervisore Magneti Marelli
Ing. Fabio Tosetto

Candidato
Sandro Gagliano
S231907

Torino, Aprile 2019

Analisi, Progettazione e Sviluppo
di Cellular-V2X Software Utilities

Sandro Gagliano

Supervisionato da:

Prof. Gianpiero Cabodi
Politecnico di Torino
Ing. Fabio Tosetto
Magneti Marelli

Sommario

Nel contesto del mondo V2X e Autonomous Driving il mio obiettivo di Tesi si suddivide in 2 differenti lavori.
Il `primo` è quello che riguarda la progettazione dell'architettura di rete necessaria alla comunicazione Veicolare, partendo dalla configurazione e gestione di un applicativo che permetta l'accesso a internet, fino a definire, tra tutti i possibili meccanismi di gestione della comunicazione, il più adatto per le esigenze aziendali. Questo livello deve infine inserirsi all'interno del **Connectivity Framework**, che è un progetto sviluppato dal gruppo di **Technology Innovation - Connectivity and Cooperative Driving Projects** in **Magneti Marelli**.
Il `secondo` obiettivo del mio lavoro di Tesi riguarda lo sviluppo di una Libreria che fornisca delle Funzioni per la gestione della connessione e connettività **LTE**. Ho implementato una serie di metodi che permettano, a qualunque entità sul Framework, la possibilità di configurare, gestire ed ottenere diverse informazioni sulla rete.

Dedica

Durante il nostro percorso di vita, qualunque siano i risultati che riusciremo ad ottenere, dovremmo sempre ringraziare le persone che ne fanno parte e che in un modo o in un altro abbiano reso possibile la realizzazione di importanti obiettivi.
Per questo motivo voglio dedicare questo piccolo spazio a chi mi ha permesso di raggiungere questo traguardo.
Ringrazio Monica, l'amore della mia vita, per il supporto che ha saputo darmi durante tutto questo tempo anche durante i momenti più difficili.
Ringrazio la mia famiglia, i miei genitori e mia sorella, che mi hanno aiutato con tutti i mezzi possibili durante tutti questi anni lontano da casa.
Ringrazio i miei amici e colleghi perchè mi hanno permesso di superare le difficoltà con una risata.
Grazie.

Indice

Elenco delle figure

Elenco delle tabelle

Lista degli Acronimi

V2X	*Vehicle-To-Everything*
V2V	*Vehicle-To-Vehicle*
V2I	*Vehicle-To-Infrastructure*
V2N	*Vehicle-To-Network*
V2D	*Vehicle-To-Device*
V2P	*Vehicle-To-Pedestrians*
V2G	*Vehicle-To-Grid*
LTE-V2X	*Long-Term Evolution Vehicle-to-Everything*
C-V2X	*Cellular Vehicle-To-Everything*
DSRC	*Dedicated short-range communications*
IVC	*Inter-Vehicular Communication*
RSU	*Road Side Unit*
OBU	*On-Board Unit*
C-ITS	*CooperativeIntelligent Transport Systems*
ETSI	*European Telecommunications Standards Institute*
FCC	*Federal Communi-cations Commission*
WAVE	*Wireless Access in Vehicular Envi-ronments*
CAN	*Controller Area Network*
UE	*User Equipment*
BS	*Base Station*

3GPP *The 3rd Generation Partnership Project*

API *Application Programming Interface*

TCP *Transmission Control Protocol*

UDP *User Datagram Protocol*

GPS *Global Positioning System*

LDM *Local Dynamic Map*

POTI *Positioning Timing*

VDP *Vehicle Data Provider*

SPAT *Signal Phase and Time*

HMI *Human Machine Interface*

BSM *Basic Safety Message*

TIM *Traveler Information Message*

WSA *WAVE Service Advertisement*

CAM *Cooperative Aware Message*

DENM *Decentralized Enviromental Notification Message*

SV *Stationary Vehicle*

FCW *Forward Collision Warning*

EEBL *Electronic Emergency Brake Ligh*

GLOSA *Green Light Optimised Speed Advise*

IMA *Intersection Movement Assist*

CLW *Control Loss Warning*

LTA *Left Turn Assist*

BSW *Blind Spot Warning*

IVRS *In-Vehicle Road Sign*

PPP *Point to Point Protocol*

Capitolo 1

Introduzione ed Obiettivi

1.1 Introduzione

Uno degli sviluppi tecnologici più importanti degli ultimi anni riguarda le reti veicolari, **Vehicular ad-hoc networks** (**VANETs**). I VANET sono stati introdotti nel 2001 [1] nell'ambito delle applicazioni Vehicle-to-Vehicle (**V2V**), dove è possibile formare reti e trasmettere informazioni tra le automobili. È ormai chiaro che le architetture di comunicazione **V2V** (Vehicle-To-Vehicle) e **V2I** (Vehicle-To-Infrastructure), coesisteranno per fornire maggiore sicurezza, navigazione e altri servizi su stradali.

Al giorno d'oggi questi servizi sono quasi diventati una necessità, l'andamento demografico, in tutto il mondo, continua a crescere e insieme ad esso anche l'urbanizzazione e la motorizzazione.

Il 2016 è stato il primo anno in cui il numero di auto prodotte, a livello mondiale, ha superato i 70 milioni (72 mila solo in Italia). Ed è proprio l'Italia ad essere tra i primi 15 paesi nel mondo ad avere un alto rapporto tra abitanti e veicoli, si stima infatti che nel nostro Paese ci siano circa 695 veicoli ogni 1000 persone [2].

Con questi dati è quindi evidente come sia necessario adottare soluzioni che permettano sia di diminuire il traffico, divenuto un enorme problema nelle grandi città come *Roma, Milano, Torino, Napoli* o *Palermo*, che il numero di incidenti.

Dai dati `ISTAT`, infatti, il numero di morti sulla strada solo in Italia, nel 2016, è salito fino a *3283* e quello di incidenti stradali a *25720*, mentre le cause principali riguardanti le circostanze di questi episodi sono da imputare a: guida distratta (16%),il mancato rispetto della precedenza o semaforo (15%) e la velocità troppo elevata (11%) [3].

Le **VANET** sono considerate quindi la soluzione adeguata per aumentare i livelli di sicurezza nelle strade tramite questo innovativo modo di comunicazione tra veicoli.
Infatti le reti veicolari sono una parte fondamentale della struttura dei sistemi di trasporto intelligenti, **Intelligent Transportation Systems** (**ITS**).

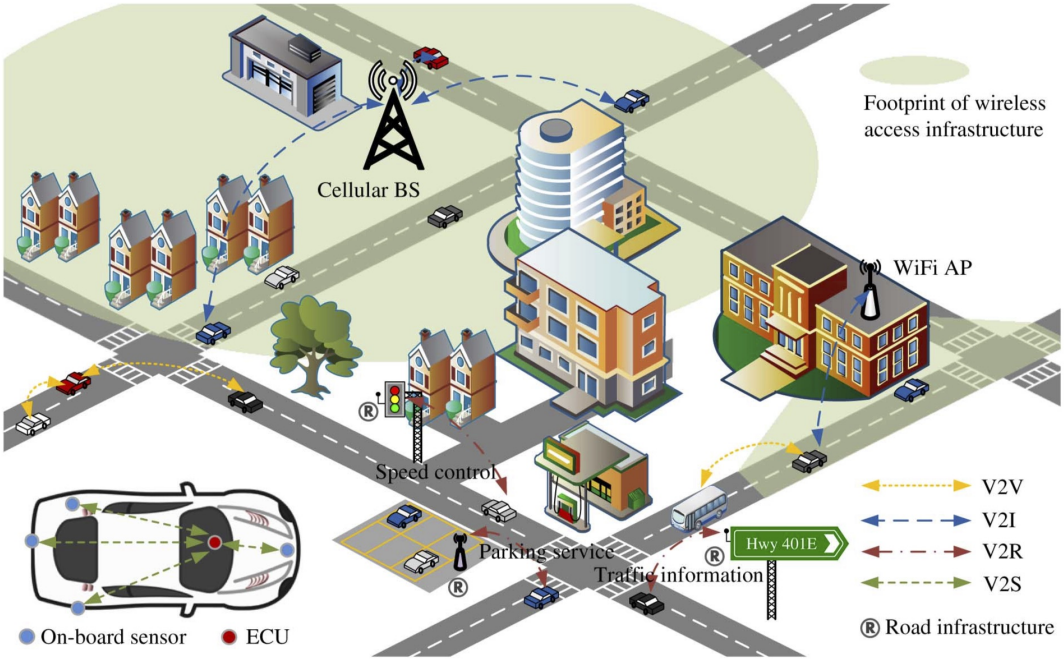

Figura 1.1: Esempio di Vehicular Ad-hoc Network (VANET)

Un **ITS** è un'applicazione innovativa che mira a fornire dei servizi avanzati relativi alle diverse modalità di trasporto e gestione del traffico ed inoltre consente ai guidatori di avere: maggiore sicurezza, più informazioni riguardo la situazione stradale in cui si trovano e più coordinati tra loro grazie all'uso delle reti di comunicazione.

I principali standard per regolare i sistemi **ITS** sono:

1. **ITS-G5**, dall'`European Telecommunications Standards Institute` (**ETSI**) in Europa;
2. **WAVE**, dall'`Institute of Electrical and Electronics Engineers` (**IEEE**) negli Stati Uniti.
3. **STD-T109**, dall'`Association of Radio Industries and Businesses` (**ARIB**) in Giappone.

In generale con il termine **V2X (Vehicle-to-Everything)** si intende un sistema di comunicazione veicolare che incorpora altri tipi più specifici di comunicazione come **V2I (Vehicle-to-Infrastructure)**, **V2N (Vehicle-to-Network)**, **V2V (Vehicle-to-Vehicle)**, **V2P (Vehicle-to-Pedestrian)**, **V2D (Vehicle-to-Device)** and **V2G (Vehicle-to-Grid)**.
Tramite meccanismi basati sul V2X si riesce a migliorare la sicurezza del traffico, migliorare la pianificazione del percorso o controllare la congestione del traffico.
In questo contesto aziende di tutto il mondo stanno sviluppando modelli software per integrare tecnologie di comunicazione wireless al mondo dei trasporti.

La **Magneti Marelli**, in particolare, sta sviluppando un **Framework V2X**, da installare nell'auto, che interagisca con gli altri veicoli e altri elementi presenti nell'infrastruttura stradale, anch'essi con il Framework installato.

Questo permetterebbe di:

- Visionare vari tipi di messaggi(allerta, pericolo, informativi etc...) all'interno del quadro strumenti per dare un supporto visivo al guidatore.
- Monitorare tutte le attività, i dati e i parametri del veicolo lato infrastruttura.
- Tracciare e Monitorare gli spostamenti dei veicoli permettendo una coordinazione veicolare utile per la gestione del traffico.

1.2 Magneti Marelli

La **Magneti Marelli** è una multinazionale italiana che opera a livello internazionale come fornitore di prodotti soluzioni e sistemi ad alta tecnologia per il mondo Automotive. La sede centrale è in Italia, a Corbetta (Milano), ed è stata fondata nel 1919, col nome di F.I.M.M. - Fabbrica Italiana Magneti Marelli, da un accordo tra Ercole Marelli e la FIAT.

Magneti Marelli fornisce tutti i maggiori produttori di automobili in Europa, Nord e Sud America e Asia.

Nel suo obiettivo di componentista automotive a livello globale,l'azienda mira a coniugare qualità e offerta competitiva, tecnologia e flessibilità, con l'obiettivo di rendere disponibili prodotti d'eccellenza a costi competitivi. Da sempre punta a valorizzare, attraverso un processo di innovazione continua, il *know-how* e le competenze trasversali di ogni dipendente al fine di sviluppare sistemi innovativi e soluzioni

che contribuiscano non solo all'evoluzione della mobilità secondo criteri di sostenibilità ambientale, ma anche a migliorare sicurezza e qualità della vita all'interno dei veicoli.
Nella `Formula 1` è stata sponsor tecnico dei principali team, tra cui `Scuderia Ferrari`, `Toyota F1 Team`, `Renault F1` e `Red Bull Racing` [4].

Magneti Marelli opera a livello internazionale attraverso otto aree di business:

- **Motorsport** (sistemi elettronici ed elettromeccanici specifici per le competizioni con leadership tecnologica in Formula1, MotoGP, SBK e WRC).
- **Automotive Lighting** (sistemi di illuminazione).
- **Plastic Components and Modules** (componenti e moduli plastici per l'automotive).
- **Suspension Systems** (sistemi sospensioni, ammortizzatori, dynamic system – sistemi di controllo dinamico del veicolo).
- **Exhaust systems** (sistemi di scarico, convertitori catalitici, silenziatori).
- **Powertrain** (sistemi controllo motore benzina, diesel e multifuel; cambio robotizzato AMT Freechoice).
- **Electronic Systems** (quadri di bordo, infotainment e telematica, lighting e body electronics).
- **Aftermarket Parts and Services** (distribuzione ricambi per l'Independent
- **Aftermarket** – IAM; (Rete Assistenza e Officine Checkstar).

1.3 Obiettivi

Nel contesto del mondo V2X e Autonomous Driving il mio obiettivo in questa Tesi si suddivide in 2 differenti lavori.

Il `primo` è quello che riguarda la progettazione dell'architettura di rete necessaria alla comunicazione Veicolare, partendo dalla configurazione e gestione di un applicativo che permetta l'accesso a internet, fino a definire, tra tutti i possibili meccanismi di gestione della comunicazione, il più adatto per le esigenze aziendali. Questo livello deve infine inserirsi all'interno del **Connectivity Framework**, che è un progetto sviluppato dal gruppo di **Technology Innovation - Connectivity and Cooperative Driving Projects** in **Magneti Marelli**.

L'obiettivo è quello infatti di sostituire i livelli più bassi dello Stack Protocollare del **TIC-Framework V2X**, che utilizza lo standard **DSRC** tramite la tecnologia `IEEE 802.11p`, con i livelli di rete definiti dallo standard `3GPP` per la comunicazione **C-V2X** (**Cellular Vehicle-to-Everything**) (Figura 1.2).

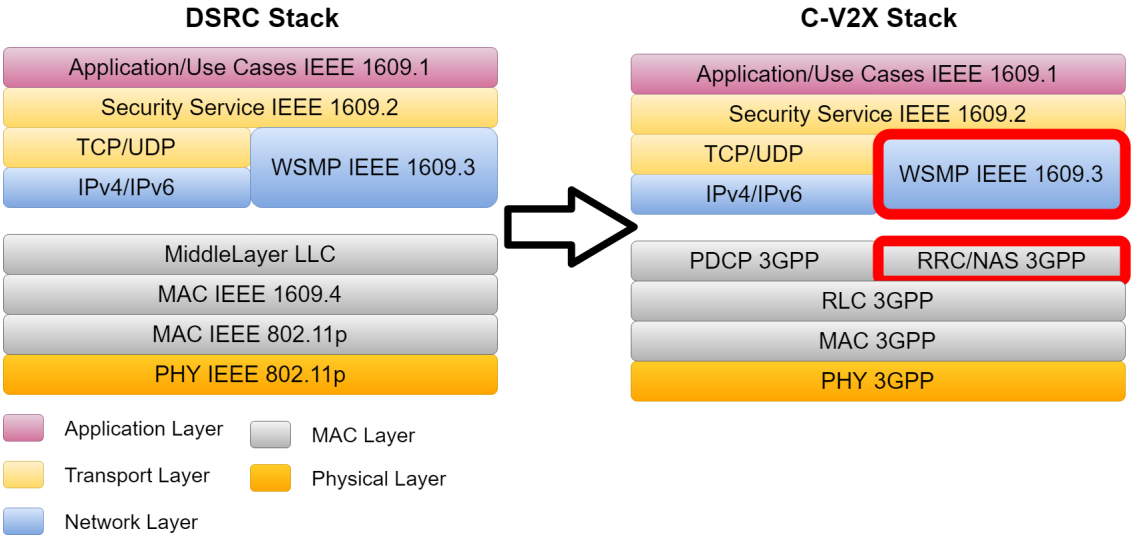

Figura 1.2: dallo Stack DSRC a quello C-V2X

La mia Tesi si inserisce all'interno del lavoro del team Magneti Marelli che in questi mesi è stato impegnato nella migrazione tecnologica del **Framework V2X** per passare da una tecnologia con **DSRC/WAVE** a una con **LTE**. Ho inserito quindi tutte le funzionalità che sono possibili integrare in un sistema Embedded, affinchè i livelli superiori del progetto continuino ad operare sempre nello stesso modo a prescindere dalla tecnologia dei livelli inferiori.

Come evidenziato in Figura 1.2, i livelli più bassi basati sul **DSRC** sono stati sostituiti dai livelli dello standard `3GPP` per le comunicazioni su rete cellulare. Le parti evidenziate in rosso sono le parti mancanti rispetto allo standard di riferimento, `3GPP Release 16`, in quanto il modem utilizzato in questa fase è conforme allo standard `3GPP Release 9` che non supporta i livelli **NAS (Non-Access Stratum)**, utilizzati per la comunicazione Broadcast.

La `seconda` parte del mio lavoro di Tesi ha riguardato lo sviluppo di una Libreria che fornisca delle **API** per la gestione della connessione e connettività **LTE**. Ho implementato una serie di funzioni che permettano, a qualunque entità sul Framework che includa e utilizzi questa libreria, la possibilità di ottenere varie informazioni sulla rete.

Lo sviluppo è stato fatto creando un modulo che riuscisse a comunicare con il modem tramite **Comandi AT**.
Questi comandi, tra le tante possibilità, permettono di ottenere dati su:

- Banda utilizzata
- Stato della connessione
- Stato di Roaming
- Tipo di connettività (`2G, 3G, 4G`)
- Informazioni sulla cella agganciata
- Informazioni sulle celle vicine
- Informazioni sul segnale (Qualità, potenza, rapporto S/N)
- Etc...

Le funzionalità non si fermano solo nella possibilità di ottenere dati utili sulla connessione ma permettono anche di configurare, scrivere o salvare informazioni valide al fine di avere il totale controllo della gestione del modem, della SIM e tutti i parametri relativi alla connettività.

In questo modo si potrà avere un maggiore controllo sullo stato della Rete e maggiori informazioni utili, per riuscire ad utilizzare la tecnologia **V2X** nel modo più efficiente possibile. Grazie a determinati meccanismi di supporto si possono inoltre effettuare analisi sulle coperture cellulari nei punti in cui si vogliono effettuare dei Test durante lo sviluppo del **Framework**, così da conoscere il contesto nel quale un determinato prototipo viene collaudato.

Capitolo 2

Dal DSRC al C-V2X

La comunicazione inter-veicolare, **Inter-Vehicular Communication (IVC)** ha conquistato l'attenzione di tutto il mondo grazie al suo principale obiettivo: migliorare la sicurezza stradale, diminuire la congestione del traffico e l'impatto del trasporto sull'ambiente.
Il loro principale vantaggio è che non richiedono la necessità di costose infrastrutture; i loro principali svantaggi sono sia i protocolli di rete relativamente complessi che la necessità di una fase di testing significativa prima che le loro applicazioni possano diventare efficaci. [5]

Le entità di un sistema IVC sono nodi che scambiano dati, l'un l'altro, in particolare sono tra Unità lato strada, **Road Side Unit (RSU)** e unità veicolo. Una RSU è fissa e montata su un lato della carreggiata, solitamente vicino al semaforo, fornisce informazioni sulla strada e comunica con veicoli dotati di un'unità di bordo, **On-Board Unit (OBU)**.

La comunicazione veicolare può operare su due diversi livelli:

- Comunicazioni a corto raggio, dove i principale standard sono **DSRC (Dedicated short-range communications)** in USA e **C-ITS (Cooperative Intelligent Transport Systems)** e in Europa

- Comunicazioni a lungo raggio, dove il principale standard è **LTE V2X (Long-Term Evolution Vehicular to Everything)**, poi rinominato successivamente **C-V2X (Cellular Vehicular to Everything)** [6]

Il mio lavoro di Tesi si inserisce in un contesto aziendale in cui il Framework V2X sviluppato da Magneti Marelli sta passando da una tecnologia basata su DSRC e standard IEEE 802.11p [7] ad una tecnologia LTE basata sullo standard 3GPP release 9-14 [6].

2.1 DSRC

Il **DSRC** è un canale di comunicazione wireless, a una o due vie, da corto a medio raggio specificamente progettati per uso automobilistico e una serie corrispondente di protocolli e standard. Negli Stati Uniti, dove è nato, la **Federal Communications Commission (FCC)** ha asseganto 75 MHZ nella banda dei 5,9 GHz . In Europa l'**European Telecommunications Standards Institute (ETSI)** ha assegnato 30 MHz di spettro nella banda dei 5,9 GHz.

Le regole di comunicazione per i dispositivi V2X sono incluse nel **SAEJ2735** Dedicated Short Range Communication (DSRC) Message Set Dictionary e comprende un set di messaggi, frame di dati e dati.
Questo standard SAE specifica un set di messaggi, i relativi frame di dati ed elementi di dati specifici per l'uso da parte delle applicazioni destinate a utilizzare le comunicazioni dedicate a corto raggio.
Sebbene lo scopo di questo standard sia incentrato su DSRC, questo set di messaggi e i relativi frame di dati ed elementi di dati sono stati progettati, per quanto possibile, anche per essere potenzialmente utilizzati per applicazioni che possono essere implementate in combinazione con altre tecnologie di comunicazione wireless.

Lo standard di comunicazione di riferimento per il **DSRC** è **IEEE 802.11p** che apporta dei miglioramenti a IEEE 802.11 (standard per il Wi-Fi) necessari per supportare le applicazioni di sistemi di trasporto intelligenti (ITS).

2.1.1 IEEE 802.11p

Lo standard IEEE 802.11 specifica un sottoscheda MAC (Medium Access Control) e diversi livelli fisici per fornire connettività wireless tra diversi nodi.

Nel caso della rete veicolare il protocollo adottato per il livello **MAC** è 802.11p, a estensione di IEEE 802.11a ottimizzando lo scambio veloce di pacchetti. Il protocollo **Medium Access Control (MAC)** in IEEE 802.11p è basato su **Enhanced distributed Channel Access (EDCA)**, meccanismo che consente a veicoli e infrastrutture di interagire l'un l'altro trasmettendo pacchetti.
Un aspetto molto importante da analizzare, tra i pacchetti scambiati, è la loro sicurezza: tutti i dati scambiati, devono essere firmati e criptati per la privacy e sicurezza dei guidatori.

2.1.2 WAVE

Il gruppo di lavoro **IEEE 1609 WAVE (Wireless Access in Vehicular Environments)** ha sviluppato la prima versione dell'architettura per i livelli inferiori dello stack protocollare specificamente adattata ai VANET [8].

Questo protocollo è stato sviluppato e pensato per lo scambio di informazioni in campo Automotive da usare sopra i livelli dell'802.11p, tuttavia è indipendente da quest'ultimi e può essere utilizzato come protocollo di riferimento anche utilizzando differenti tecnologie wireless.
Uno degli aspetti più interessanti è infatti la possibilità, tramite minime modifiche, di cambiare la tecnologia wireless sulla quale si basa il DSRC mantenendo intatto tutto lo standard WAVE. Questo è uno dei punti di forza che sta permettendo la migrazione dei sistemi V2X basati su 802.11p verso sistemi che utilizzino tecnologie GSM o LTE nei livelli più bassi della pila protocollare.

Lo standard IEEE WAVE, descritto nel documento 1609, è diviso in quattro parti, come si può vedere dalla Figura 2.1 che mostra la posizione di ogni parte dello standard nello stack protocollare.

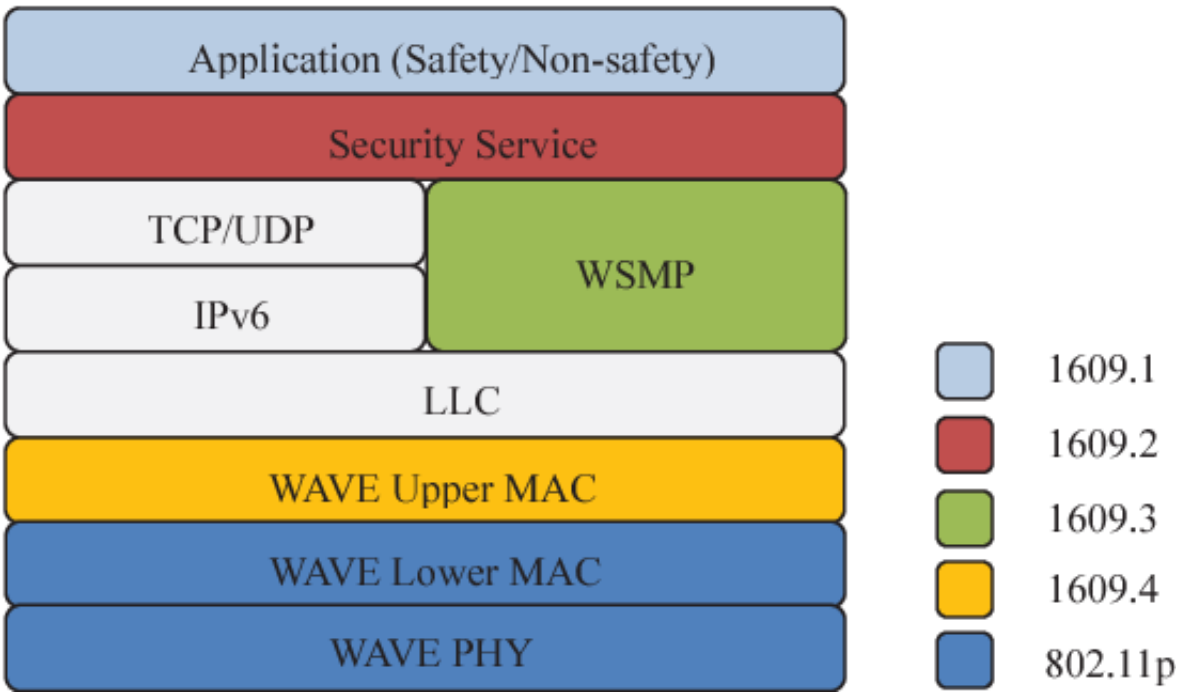

Figura 2.1: Architettura Stack per il modello DSRC/WAVE usato nelle applicazioni V2X

I Quattro standard **IEEE** relativi ala comunicazione Wave sono:

- **1609.1** WAVE Resource Manager (RM), specifica in che modo le applicazioni nei siti remoti possono comunicare con le OBU tramite le RSU; Si occupa anche di come saranno il formato dei messaggi di comando e il formato di archiviazione dei dati, in modo che questo possa essere molto utile al livello applicativo per comunicare tra i componenti o qualche dispositivo speciale supportato dall'OBU.

- **1609.2**, specifica i servizi di sicurezza e le regole da applicare nello standard WAVE. Il servizio di gestione della sicurezza fornisce un servizio di gestione dei certificati fornito da Certificate Management Entity (CME). In generale supporta alcuni servizi di sicurezza generici come riservatezza, autenticazione, autorizzazione e integrità.

- **1609.3**, Specifica il servizio di rete e di trasporto. Per supportare lo scambio sicuro di dati WAVE, il servizio di rete definisce i servizi di trasporto e di rete, oltre a includere l'indirizzamento e il modo in cui avviene il routing. I servizi di rete includono anche il controllo del Logical Link Layer, per il traffico IP e WSMP (WAVE Short Message Protocol). Questo ambito è una specifica del livello di trasporto e del livello di rete che supporta la connettività wireless multicanale in WAVE [9].

- **1609.4**, specifica le operazioni di accesso multicanale. Questo standard fornisce fondamentalmente il miglioramento del controllo di accesso ai media IEEE 802.11 per supportare il funzionamento delle onde e i servizi che supportano i servizi multicanale nell'ambiente WAVE [10]. Questo standard è utilizzato per gestire la coordinazione dei canali (coordinate del livello MAC che il pacchetto venga trasmesso nel canale RF corretto al momento giusto) e supportare l'invio del pacchetto dati del servizio MAC.

Esistono due classi di dispositivi in un sistema WAVE:

- L'unità di bordo **(OBU)**, On-Board Unit.

- L'unità a bordo strada **(RSU)**, Road Side Unit.

Queste sono rispettivamente equivalenti alla Mobile-Station (MS) e alla Base-Station (BS) nei sistemi cellulari.

Ci sono due classi di comunicazioni possibili da OBU e RSU: da veicolo a veicolo (V2V) e veicolo all'infrastruttura (V2I).
Mentre una nel mondo cellulare un MS normalmente comunica con un'altra MS tramite la BS, l'OBU in un veicolo comunica direttamente con altri OBU all'interno dell'area di copertura radio. Questo V2V diretto riduce la latenza dei messaggi; la

bassa latenza è un requisito essenziale per le applicazioni di sicurezza poiché permette una migliore gestione nella prevenzione delle collisioni tra automobili.

Un altro aspetto da mettere in evidenza è che un OBU è incorporato e connesso con altri sistemi di dispositivi elettronici del veicolo tramite reti interne come la **CAN (Controller Area Network)**.
Inoltre migliorando la sicurezza, le reti WAVE possono svolgere ruoli importanti nel piano di viaggio, nella gestione del traffico, navigazione, monitoraggio ambientale, logistica, congestione e riduzione delle emissioni, riscossione dei pedaggi, parcheggio intelligente, servizi di emergenza e una vasta gamma di altri servizi basati sulla posizione.

Le reti WAVE hanno una serie di sfide tecniche che non si incontrano in altre reti wireless. Una sfida consiste nell'utilizzare la tecnologia WAVE nella prevenzione degli incidenti tra veicoli in movimento. Ad esempio, può essere usato per avvisare i conducenti ad un incrocio della presenza di un altro veicolo al fine di evitare la collisione. Nella comunicazione V2V il sistema deve essere robusto anche in situazioni estremamente anormali, come incidenti e collisioni. Un esempio di una situazione anormale si ha, per esempio, quando due auto viaggiano su una stretta strada a doppio senso l'una verso l'altra a velocità elevate.
Fondamentalmente le reti WAVE devono essere estremamente solide altrimenti un loro fallimento potrebbe causare, nei casi peggiori di un incidente, la perdita della vita del conducente.
Le Reti WAVE devono essere scalabili, robuste, a bassa latenza e alta capacità.

2.2 C-V2X

Cellular Vehicle-to-Everything (C-V2X) è uno standard di connettività unificata progettato per offrire bassa latenza nelle comunicazioni Vehicle-To-Vehicle (V2V), Vehicle-To-Infrastructure (V2I) e Vehicle-To-Pedestrian (V2P), senza però dimenticare di mantenere l'affidabilità e la robustezza nelle comunicazioni Vehicle-To-Network (V2N). Collegando i singoli veicoli e consentendo lo sviluppo di sistemi di trasporto intelligenti cooperativi (C-ITS) che riducono la congestione e l'inquinamento, la piattaforma ha il potenziale per trasformare le informazioni e i servizi di sicurezza sulle autostrade e nelle città per migliorare i viaggi.

C-V2X è definito come LTE-V2X nello standard 3GPP Release 14 [6], poi rinominato nella Release 15 come C-V2X dove le funzionalità V2X vengono espanse per supportare 5G.

Il **C-V2X** è progettato per funzionare in 2 diverse modalità:

- **Comunicazione diretta**: tramite l'interfaccia **PC5**, usata per far comunicare i dispositivi (OBU, RSU) tra loro (V2V, V2I, V2P)
- **Comunicazione di rete**: tramite l'interfaccia **Uu**, usata per comunicare le informazioni dal/al veicolo tramite la Rete Cellulare (V2N);

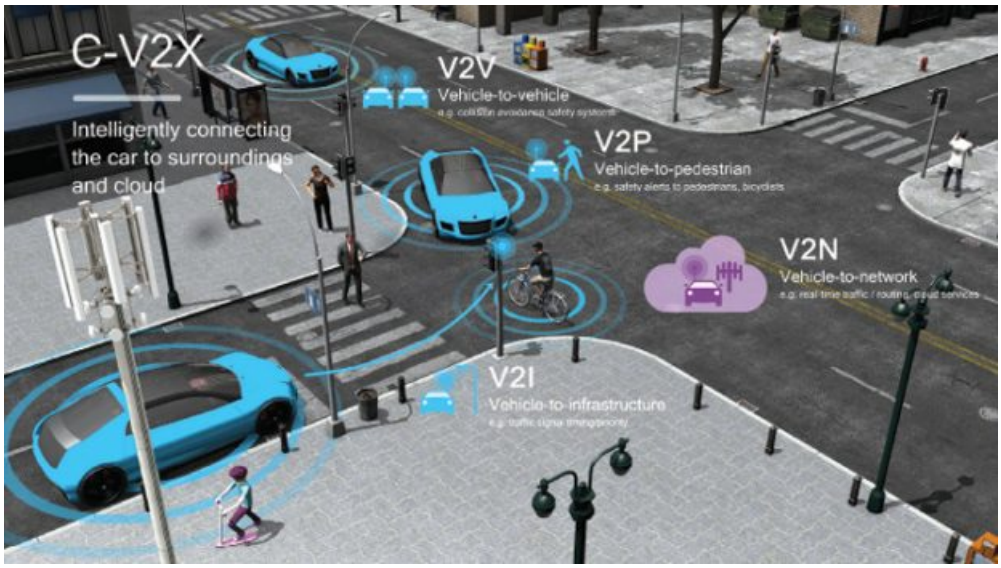

Figura 2.2: Esempi di comunicazione nel C-V2X, nel quale tutti i tipi di connessioni (V2V, V2I, V2P, V2N) convivono all'interno della stessa architettura [11]

2.2.1 Interfaccia PC5

La comunicazione diretta tra veicolo e altri dispositivi utilizza la cosiddetta interfaccia **PC5**.
PC5 fa riferimento a un punto di riferimento in cui l'**UE** (User Equipment), ovvero il telefono mobile, comunica direttamente con un'altra UE sul canale diretto. In questo caso la comunicazione con la Base Station (BS) non è richiesta.
Questa possibilità di comunicazione diretta permette una maggiore gestione del traffico dati, che così non deve passare da terzi per effettuare la consegna dell'informazione, riducendo enormemente la latenza.

L'interfaccia PC5 è nata per soddisfare le esigenze di comunicazione per la pubblica sicurezza [12]. Questa necessità è nata per consentire alle forze dell'ordine o al soccorso di emergenza di utilizzare la comunicazione su rete cellulare anche quando l'infrastruttura di rete non è disponibile, come nello scenario di calamità naturali.

2.2.2 Interfaccia Uu

La comunicazione di rete utilizza la connessione di rete cellulare in modo tradizionale tramite l'interfaccia **Uu**.
Uu si riferisce all'interfaccia logica tra UE e il BS ed è generalmente utilizzata per il **V2N**. Il V2N è un caso di utilizzo unico per il C-V2X e non esiste nel V2X basato su 802.11p dato che quest'ultimo supporta solo la comunicazione diretta in broadcast.

Per utilizzare l'interfaccia Uu è quindi necessaria la registrazione ad una cella che autentichi e consenta la connessione del veicolo introducendolo nella rete. Nel V2N il veicolo comunica quindi le informazioni relative al proprio stato (velocità, direzione, messaggi di pericolo etc...) ad un nodo raggiungibile tramite rete internet che funge da server e, tramite opportune logiche, invio queste informazioni ad altri veicoli o ad altri dispositivi che utilizzano questi dati per effettuare varie operazioni come il tracking del veicolo o la gestione del traffico.

Le due interfacce convivono all'interno dello stesso progetto per consentire un migliore utilizzo del mezzo trasmissivo, che nel caso del V2V diretto non deve subire latenze dovute al dover comunicare prima con un nodo intermedio, ampliando allo stesso tempo il range di comunicazione tramite il V2N.

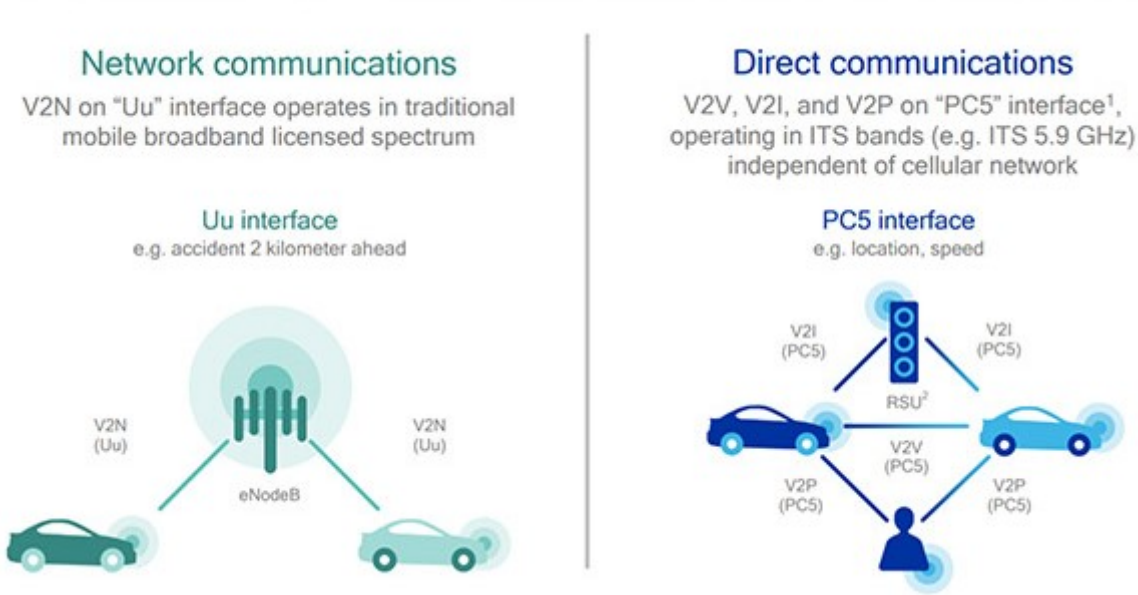

Figura 2.3: Esempi di comunicazione veicolare tramite le interfacce Uu (a sinistra) e PC5 (a destra) [13]

2.3 Differenze tra il DSRC e il C-V2x

Le differenze tra DSRC e C-V2X si estendono in diversi ambiti, dagli aspetti Tecnici alle proprietà generali. Le Figure 2.4 e 2.5 evidenziano le principali dissomiglianze e similitudini tra i due Standard.

Technology operation	802.11p	C-V2X Rel-14/15	C-V2X Rel-16 (expected design)
Specification completed	Completed	Rel-14 completed in 2016. Rel-15 to be completed in 2018	2019
Support for low latency direct communications	✓	✓ (Rel-14 – 4ms)	✓ (≤ 1ms)
Support for network communications	Limited (via APs only)	✓	✓
Can operate without network assistance	✓	✓	✓
Can operate in ITS 5.9 GHz spectrum	✓	✓	✓
SIM-less operation	✓	✓	✓
Security and privacy on V2V/V2I/V2P	✓ (as per IEEE WAVE and ETSI-ITS security services)	✓ (as per IEEE WAVE and ETSI-ITS security services)	✓ (as per IEEE WAVE and ETSI-ITS security services)
Security/Privacy on V2N	N/A	✓	✓
Coexistence in 5.9GHz	✓ (Adjacent channel with 3GPP tech)	✓ (Adjacent channel with 11p; co-channel coexistence from R14 onwards)	✓ (Adjacent channel with 11p; co-channel coexistence from R14 onwards & WiFi)
Evolution path	✗	✓	✓ Compatible with Rel-14/15

Figura 2.4: Confronto Tecnico tra DSRC e C-V2X [13]

Negli ultimi anni, il DSRC è stata l'unica tecnologia V2X disponibile. Dopo un lungo periodo di test sul campo e su vasta scala, il V2X basato su DSRC è entrato in produzione negli Stati Uniti e in Giappone nel 2017 e sta per iniziare la produzione di massa in Europa nel 2019. Recentemente, è stata introdotta la tecnologia C-V2X, avendo lo stesso scopo di comunicazione diretta tra veicoli.

Il C-V2X è definito dal **3GPP (The 3rd Generation Partnership Project)**, basato sulla tecnologia dei modem cellulari, porta i livelli di accesso totalmente diversi e non interoperabili con DSRC. A parte questo, le due tecnologie messe a confronto stanno affrontando casi d'uso identici e hanno livelli simili di rete, sicurezza e d'applicazione.

DSRC e C-V2X sono basati su tecnologie diverse, che portano a metodi operativi fondamentalmente diversi. DSRC, derivato da WiFi, è ottimizzato per costi e semplicità e supporta intrinsecamente operazioni distribuite. C-V2X, derivato da LTE, ha aggiunto nuovi meccanismi per abilitare l'operazione distribuita.

Guardando invece le performance, Figura 2.5, è possibile vedere gli enormi vantaggi che la tecnologia C-V2X può dare rispetto alla concorrenza. Notiamo come, mentre nel DSRC la perdita dei pacchetti (Packet Loss) è bassa in condizioni di alta densità di veicoli, nel C-V2X riusciamo addirittura ad ottenere una garanzia di invio su ogni singolo pacchetto, annullando la perdita del dato anche nelle condizioni più estreme.

Il vantaggio più grande lo si può notare anche nel Range di Trasmissione della comunicazione diretta (V2V, V2I, V2P) che passa dai 225 metri del DSRC agli oltre 450 metri del C-V2X, mantenendo tempi di latenza inferiori all' 1 ms (3GPP Release 16). Inoltre grazie all'infrastruttura di rete cellulare è finalmente possibile ottenere un V2N, quindi un accesso alla rete, per riuscire ad avere una comunicazione a distanza tramite rete cellulare.

Use Cases	802.11p	C-V2X Rel-14/15	C-V2X Rel-16(expected design)
Target Use Cases	Day 1 safety only	Day 1 safety & enhanced safety use cases	Advanced use cases to assist in autonomous driving including, ranging assisted positioning, high throughput sensor sharing & local 3D HD map updates
Performance			
High density support	✓ Packet loss at high densities	✓ Can guarantee no packet loss at high densities	✓ Can guarantee no packet loss at high densities
High mobility support	✓ Up to relative speeds of 500 km/hr with advanced receiver implementation	✓ Up to relative speed of 500 km/hr as a minimum requirement.	✓ Up to relative speed of 500 km/hr as a minimum requirement
Transmission range @ 90% error, 280 km/hr relative speed	Up to ~225m	-Over 450m using direct mode -Very large via cellular infrastructure	-Over 450m using direct mode -Very large via cellular infrastructure
Typical transmission frequency for periodic traffic	Once every 100msec (50ms is also possible)	Once every 100ms (20ms is also possible)	Supports packet periodicities of a few ms.

Figura 2.5: Confronto sui Casi d'Uso e performance tra DSRC e C-V2X [13]

L'enorme differenza la si può poi sperimentare, oltre che dal lato tecnico, anche dal punto di vista dell'utilizzo, infatti un esempio molto più tangibile è quello relativo agli **Use Cases (Casi d'Uso)**.
Ogni veicolo, sia che esso sia fornito dell'una o dell'altra tecnologia, a livello applicativo implementa Use Case quasi identici che però subiscono enormi cambiamenti sui servizi che riescono ad offrire.

Uno Use Case non è altro che un applicativo di alto livello che offre dei servizi o funzionalità basati sui dati che riceve dall'ambiente, o da altri veicoli.
Uno degli Use Case più famosi, Fig. 2.6, è il caso del pedone dietro la curva che non può essere visto né dal guidatore né dai sistemi oggi in uso come radar, Lidar e telecamere. Grazie al mondo C-V2X avrà una possibilità di essere visto grazie al fatto che lo smartphone comunica la sua posizione a tutti i veicoli sulla strada.
Lo stesso caso può essere visto anche nel DSRC con l'importante differenza che l'informazione potrà essere fornita solo a partire da una certa distanza dalla fonte.
Il caso del pedone vale ovviamente anche per auto/moto/camion e ciclisti, in caso anche di sorpasso, cambio di corsia, svolte e intersezioni di ogni genere.

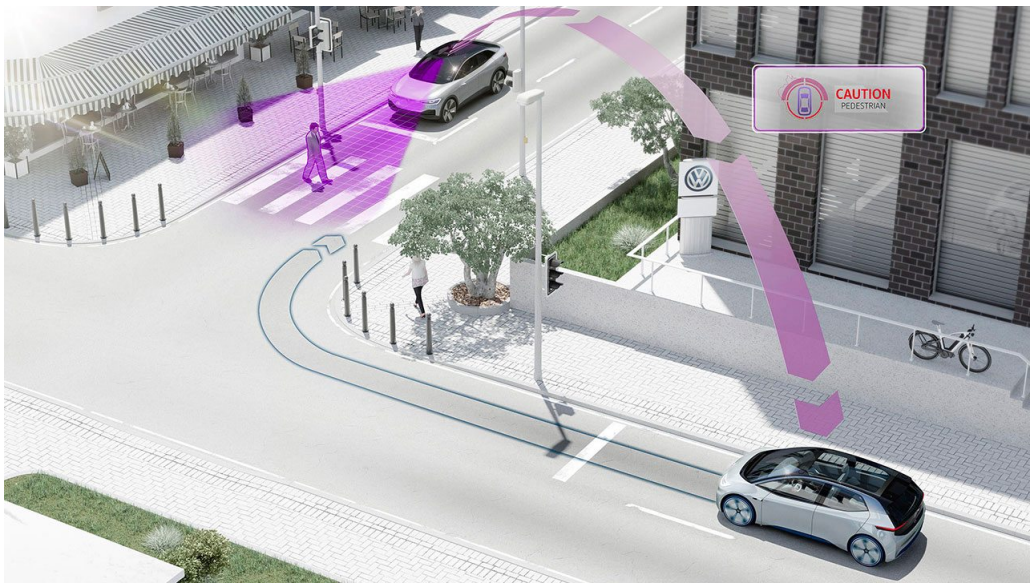

Figura 2.6: Use Case sul pedone e il punto cieco nel mondo C-V2X [14]

2.4 Migrazione dal DSRC al C-V2X

Nonostante i due Standard siano molto diversi è possibile riutilizzare alcuni moduli del mondo DSRC per essere integrati nel mondo C-V2X.
Come mostrato in Figura 2.7, cambiando i moduli dei livelli 1 e 2 (riferiti al modello ISO/OSI) con quelli 3GPP come standard su rete cellulare, è possibile riutilizzare tutta l'architettura software precedentemente sviluppata.

Sviluppando quindi un nuovo livello relativo alla nuova tecnologia da utilizzare a livello fisico, modem cellulare (2G, 3G, 4G, 5G), è possibile sviluppare un nuovo Framework mantenendo tutti i moduli di alto livello già presenti.

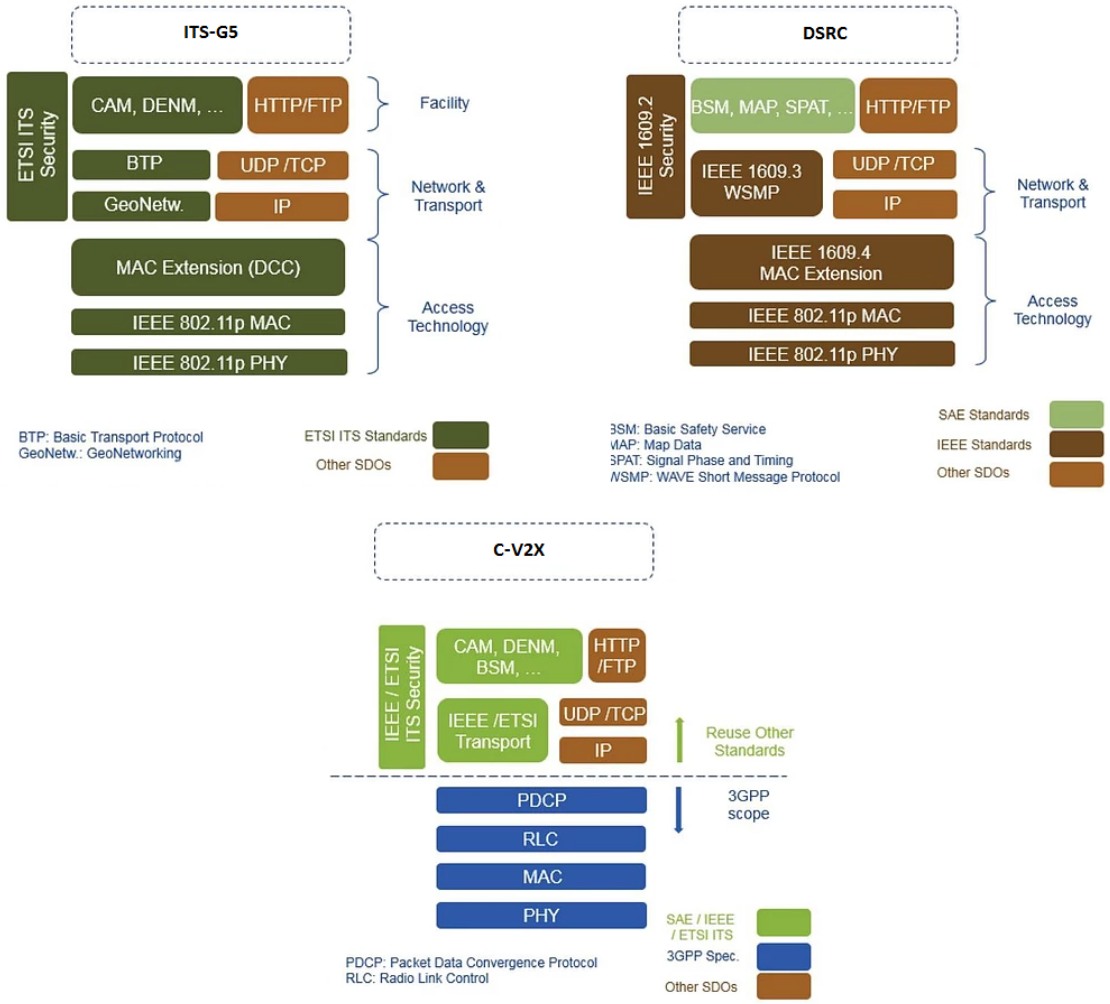

Figura 2.7: Differenze tra gli Stack del ITS-G5, DSRC, C-V2X [15]

Il team di sviluppo di **Technology Innovation - Connectivity and Cooperative Driving Projects** in **Magneti Marelli**, si è quindi impegnato in questa fase di migrazione del TIC-Framework da una tecnologia Wi-Fi basata su IEEE 802.11p ad una di telefonia mobile basata su standard 3GPP.

Purtroppo, però, la lentezza del mercato ha impedito una migrazione completa al mondo C-V2X che dalle proiezioni dovrebbe iniziare i primi test nel 2020. Infatti gli ultimi Modem non implementano ancora l'interfaccia PC5, necessaria per la parte di comunicazione V2V, V2I, V2D, V2P. Il team di sviluppo ha quindi deciso di adottare soluzioni alternative per iniziare la prima fase di questa migrazione anche senza l'interfaccia PC5.

Capitolo 3

Long-Range V2X

3.1 Introduzione

I primi Test per la comunicazione tramite lo standard **C-V2X** sono già in corso e per il 2020 si farà riferimento al `3GPP Release 16` come requisito minimo.
Nel progetto **TIC-Framework** il modem utilizzato è **Telit-LE920-EU**, il cui standard di riferimento è il `3GPP Release 9`. Mancando, quindi, la componente **PC5**, come evidenziato nel Paragrafo 2.4, si è optato per una soluzione intermedia tra il **DSRC** e il **C-V2X**, chiamata dal team di sviluppo **Long-Range V2X**.

L'architettura di questa particolare versione può essere vista in Figura 1.2, e rimedia alla mancanza del PC5 utilizzando una topologia di rete a centro stella, Figura 3.1. Questa topologia di rete consiste in una entità che funge da punto centrale per la trasmissione delle informazioni e ogni host è connesso logicamente a tale punto.

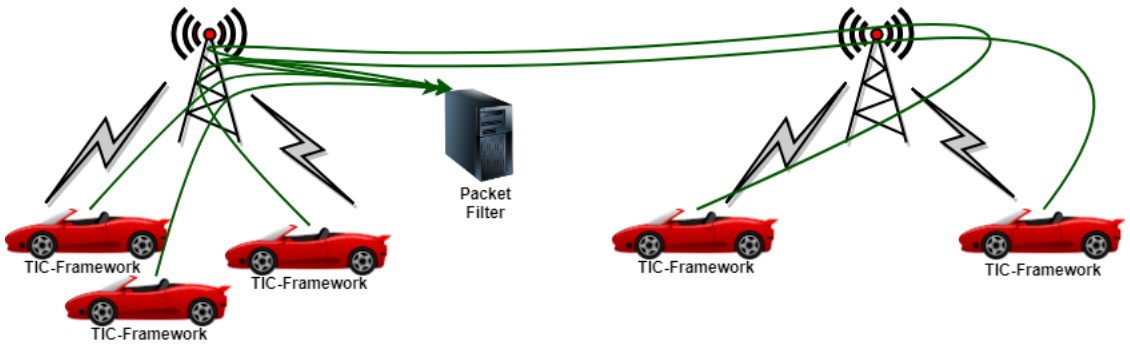

Figura 3.1: Esempio del flusso Dati nel Long-Range V2X

I dati all'interno di una rete a stella attraversano un nodo intermedio prima di arrivare a destinazione. Questo nodo oltre al Framework è dotato di un modulo sviluppato per eseguire le operazioni di smistamento chiamato **Packet Filter**.

3.2 Topologia a Stella

La rete a stella è una rete locale caratterizzata dalla presenza di un mainframe centrale, generalmente di grandi dimensioni, a cui sono collegati, fisicamente o logicamente, diversi host. Il nodo centrale ripartisce le risorse del sistema a partizione di tempo fra tutti i nodi associando a tutti gli utenti una frazione di secondo in modo ciclico.

3.2.1 Vantaggi

I vantaggi di questa struttura sono:

- **Isolamento dell'errore**: quando il collegamento tra il Packet Filter e una Board cade, per motivi legati alla scarsa qualità del segnale, ridotta copertura dell'host o semplicemente un bug del software presente nella Board, le funzionalità della rete restano invariate e tutti gli altri host possono continuare a comunicare tra loro grazie al Packet Filter.

- **Basso Costo**: creare una rete con topologia a stella permette di investire le principali risorse nel centro stella che è il punto più importante di tutta la rete, concentrando lo sviluppo nel cercare di renderlo più robusto possibile.

- **Semplice Implementazione e Gestione**: creare una rete con topologia a stella è piuttosto semplice, soprattutto rispetto ad altre reti molto più articolate. La gestione di questa rete si concentra principalmente nella gestione del centro stella.

3.2.2 Svantaggi

Gli svantaggi di questa struttura sono:

- **Simulazione del Broadcast**: quando una Board deve inviare un messaggio sulla rete, il messaggio deve passare prima per il nodo centrale e dopo essere inviato a tutte le Board che si sono registrate presso il Packet Filter. Questo nasce per avere la necessità di recapitare il messaggio a tutti i veicoli vicini.

- **Bottleneck** (Collo di Bottiglia): se il traffico lungo la rete diventa particolarmente intenso, la velocità di trasmissione può diventare molto bassa, in quanto le informazioni devono passare, come già si è detto, dal nodo centrale

- **Single Point of Failure**: se il Packet Filter ha un guasto, un bug o un qualunque problema (sia Hardware che Software) l'intera rete rimane paralizzata e i veicoli non riescono più a comunicare tra loro.

3.3 Architettura Long-Range V2X

L'obiettivo del gruppo di lavoro è quello di aver effettuato entro il 2020 una migrazione completa del **TIC-Framework V2X** al mondo 5G. Una tappa intermedia obbligatoria però risulta essere questa versione soprannominata **Long-Range V2X**. A causa della mancanza della tecnologia PC5, che sarà disponibile con i modem di nuova generazione, una delle componenti, forse, più importanti del progetto C-V2X non può essere completata.

L'interfaccia **PC5**, come discusso nella Sezione 2.2.1, è quella componente che permette di utilizzare la comunicazione **V2V**, quel tipo di comunicazione diretta che permette ai veicoli di scambiarsi i pacchetti informativi senza dover passare dalla rete internet.

L'opzione che gli sviluppatori di tutto il mondo stanno adottando è quindi quella di simulare un "finto" V2V. Questo V2V simulato permette agli applicativi di funzionare pensando di essere ancora in un'architettura classica del V2X dove la comunicazione veicolare esiste, mentre invece a livello di rete questa comunicazione avviene tramite indirizzi IP, quindi attraverso la rete.

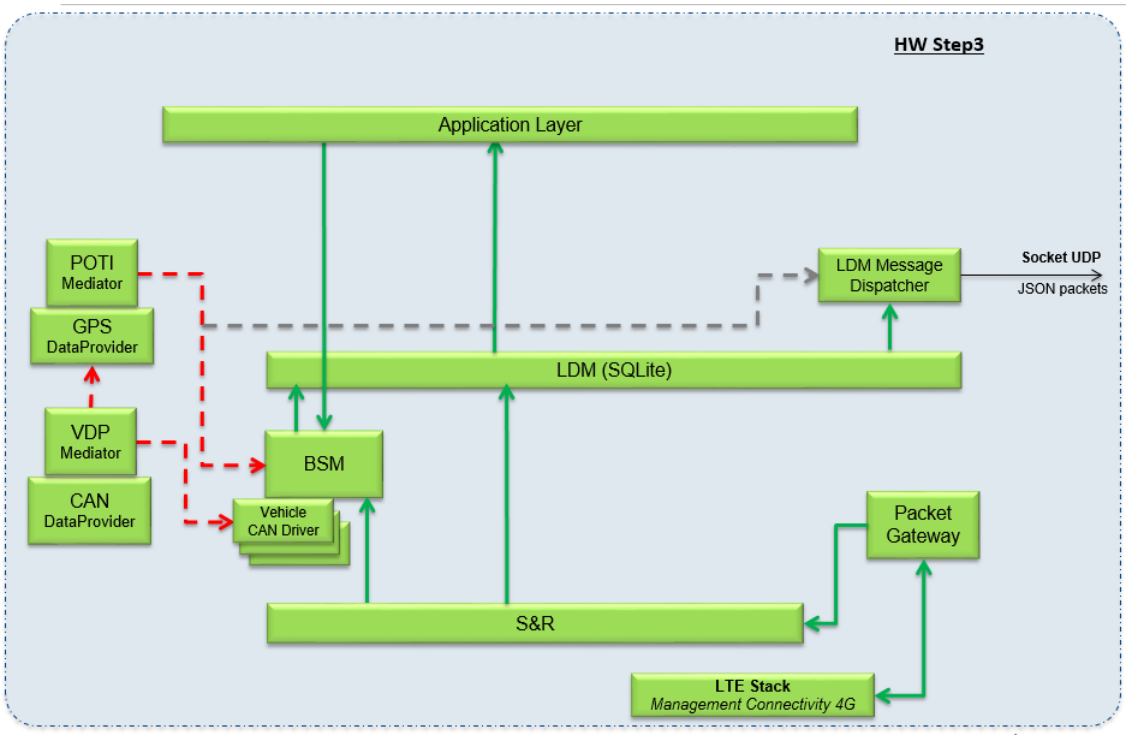

Figura 3.2: Architettura Packet Filter nel Long Range V2X

3.4 Differenze DSRC, C-V2X e Long-Range V2X

Di seguito una tabella riassuntiva delle differenze più importanti tra i due Standard di riferimento, **DSRC** e **C-V2X** e questa versione intermedia chiamata **Long-Range V2X**.

Specification	**DSRC**	**C-V2X**	**Long-Range V2X**
Standard Network	IEEE 802.11p	3GPP Rel.16	3GPP Rel.9
Protocol Application	IEEE	IEEE/ETSI	IEEE
Frequency Range	5.850 - 5.925 GHz	5.850 - 5.905 GHz	5.850 - 5.905 GHz
V2V Communication	WAVE	PC5	Not Available
Topology Network	Broadcast	Broadcast	Star
Default MAC Support	802.11p	4G/5G	4G/5G

Tabella 3.1: Paragone tra le due tecnologie di riferimento e il Long-Range V2X.

Come è evidente dalla Tabella 3.1, le principali differenze con i due standard si possono suddividere su 3 grandi ambiti:

- **Protocolli Applicativi**: gli Use Cases e tutti i moduli del livello Applicativo nel mondo C-V2X possono essere sia appartenenti agli standard ETSI che a quelli IEEE/WAVE. Grazie al Protocollo 3GPP è definito un layer intermedio che funge da middleware per l'invio in Broadcast dei messaggi dei due Standard. Al momento attuale il Long-Range V2X include solo il mondo WAVE ereditato dal progetto del Framework in DSRC. Nell'immediato futuro è prevista l'integrazione del mondo ETSI.

- **Comunicazione Intra Veicolare**: come già detto ed evidenziato in Tabella, il Long-Range V2X compensa la mancanza di un'interfaccia dedicata alla comunicazione Broadcast utilizzando una topologia a stella tramite l'inserimento di un modulo chiamato Packet Filter. Grazie a questa possibilità la comunicazione V2V viene ottenuta utilizzando la copertura cellulare e mandando i dati sulla rete Internet.

- **Standard Livello di Rete**: il livello di rete è ovviamente il principale layer che è cambiato nel passaggio dal DSRC. Non si usa più il Wifi ma tutta la comunicazione avviene su Rete Mobile. Tuttavia lo scope del C-V2X viene standardizzato nella Release 16 del 3GPP. Attualmente con il Modem attuale viene assicurato uno standard 3GPP Rel.9, più che sufficiente per i test che si stanno eseguendo all'interno del gruppo di Technology Innovation di Magneti Marelli.

3.5 Motivazioni

La necessità di sviluppare questa versione intermedia chiamata **Long-Range V2X** nasce dal crescente sviluppo che le tecnologie basate su V2X stanno avendo in questi anni. Nonostante le difficoltà nello sviluppo della versione finale del C-V2X tutti stanno cercando di non ritrovarsi impreparati per l'arrivo di questo nuovo standard.

La mancanza dell'interfaccia PC5 è solo un piccolo tassello all'interno di quello che il C-V2X punta a diventare. Grazie a queste soluzioni intermedie possono essere risolti i primi problemi che di solito accompagnano ogni innovazione.

I primi Test confermano infatti quello che si pensava. Con il C-V2X si può riuscire ad ottenere un enorme balzo in avanti, maggiore velocità di comunicazione, ampliamento del range di trasmissione, miglioramento dell'infrastruttura e soprattutto un sistema che può essere la base solida su cui fare affidamento per le tecnologie che riguardano il Veicolo Autonomo.

Capitolo 4

Hardware e Software Magneti Marelli

4.1 Introduzione

In questo capitolo sono riportati alcuni concetti relativi allo sviluppo del gruppo di **Technology Innovation - Connectivity and Cooperative Driving Projects** in **Magneti Marelli** sul progetto per il V2X. Inoltre, sono riportati i dettagli sull'hardware e le funzionalità software utilizzate nel progetto **TIC-Framework (Technology Innovation Connectivity Framework)**.

4.2 Tecnologie Wireless utilizzate

All'interno del mio elaborato di Tesi ho avuto modo di affrontare e studiare due tipi di tecnologie nel progetto del V2X:

- **IEEE 802.11p / DSRC**: questo standard IEEE specifica le frequenze Wireless, i tempi, i maccanismi di accesso alla rete ecc. per abilitare le comunicazioni dedicate a corto raggio. Sono inclusi anche gli standard IEEE 1609 (dal .1 al .4) che definiscono il protocollo delle WAVE e le funzionalità di sicurezza e comunicazione diretta.

- **C-V2X / LTE-V2X**: questo standard si basa sulle tecnologie di comunicazione cellulare **4G** e **5G** per fornire applicazioni in un range più ampio. Introduce la possibilità del V2N, permettendo ai dati del veicolo di viaggiare nella rete Internet.

4.3 Magneti Marelli Connectivity Framework

Il gruppo **Technology Innovation - Connectivity and Cooperative Driving Projects** divisione della Magneti Marelli sta sviluppando un software per il trattamento delle comunicazioni V2X tramite DSRC, C-V2X e conforme agli standard WAVE ed ETSI. Di seguito la struttura del Connectivity Framework Magenti Marelli caratterizzata da tre livelli: Middleware, Facilities e Applications/Use Cases. L'architettura del framework è riportata in figura 4.1.

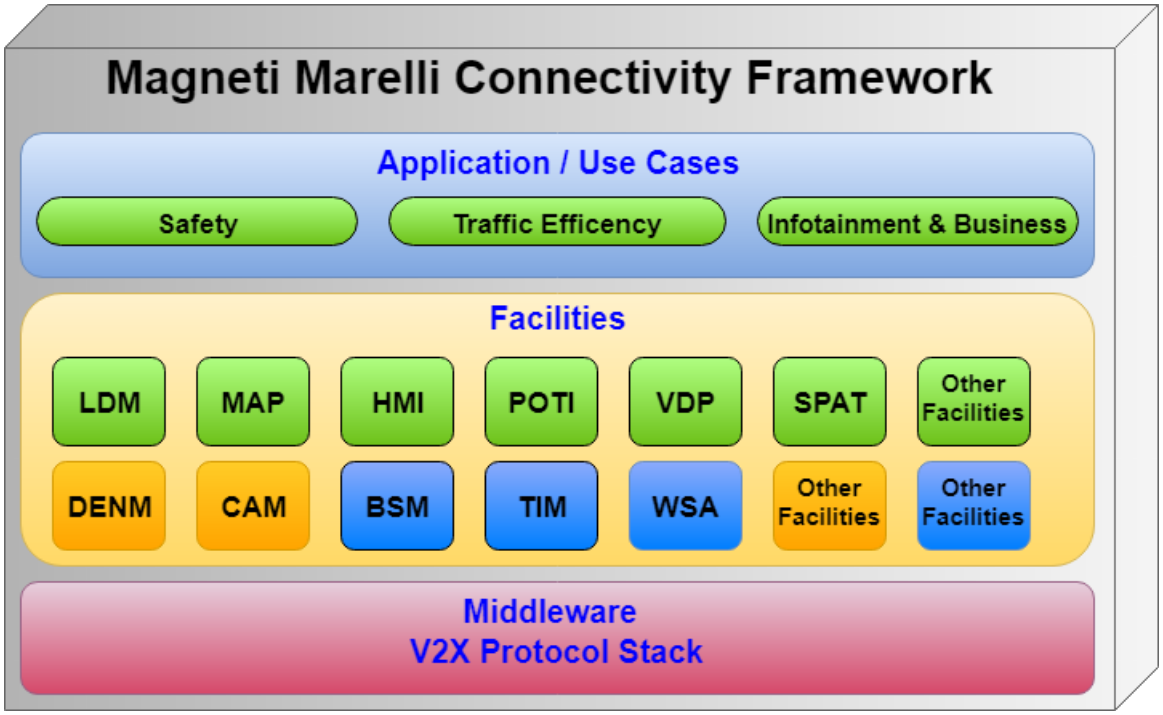

Figura 4.1: Architettura Framework

4.3.1 Middleware Protocol Stack

Questo livello fornisce supporto per le diverse tecnologie di connettività utilizzate dalle applicazioni V2X, come 802.11p o le varie versioni di C-V2X. Driver e librerie per il trasferimento di informazioni da e verso i dispositivi utilizzati dal Framework sono posizionati in questo livello.
Qui si inserisce il mio lavoro di Tesi nel quale è stato necessario passare da una configurazione dello stack di Rete pensata per l'802.11p ad uno Stack LTE per fornire connettività cellulare.

4.3.2 Facilities

Questo strato contiene un insieme di moduli software e definizioni di dati (come i messaggi) che forniscono il supporto richiesto per i casi d'uso, Use Cases. Ogni entità può essere usata in entrambi gli standard ETSI e WAVE.

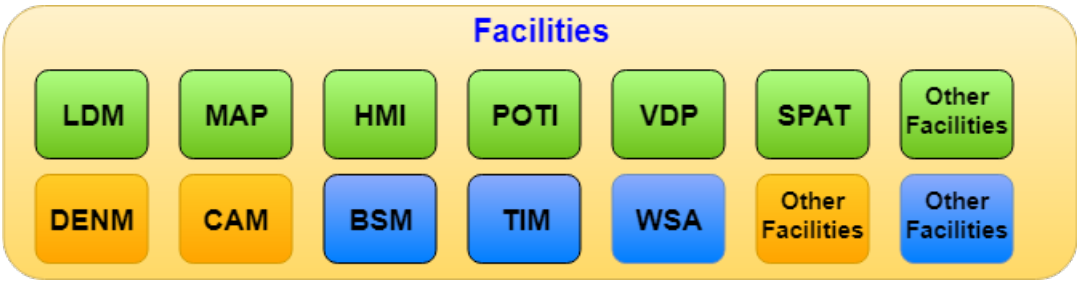

Figura 4.2: Facilities Framework

I Servizi comuni ad entrambi gli standard **WAVE** ed **ETSI**, evidenziati in figura 4.2 sono:

- **LDM (Local Dynamic Map)**: questo componente funziona come un database che memorizza dati rilevanti ricevuti da veicoli, infrastrutture, ecc. La sua Funzione è quella di gestire le informazioni del traffico stradale, acquisendo dati da infrastrutture esterne, veicoli, segnaletiche. Il modulo LDM è un elemento fondamentale del Connectivity Framework, dal quale passano la maggior parte delle informazioni.
- **POTI (Positioning Timing)**: questa entità fornisce informazioni sulla posizione del veicolo come latitudine, longitudine e altitudine. Il POTI legge i dati ricevute dai sensori, ad esempio il GPS, e a partire da questi compone il pacchetto informativo con dati sulla posizione della stazione mobile. Si occupa inoltre di sincronizzare il timing del sistema.
- **VDP (Vehicle Data Provider)**: componente accoppiato con il POTI, fornisce ai componenti deli livelli superiori altre informazioni sul veicolo come la sua dinamica (proveniente dalla rete CAN).
- **MAP (Topology Message)**: un componente che gestisce un tipo specifico di messaggio, chiamato per l'appunto MAP. Questo messaggio contiene informazioni geografiche dettagliate sulla strada attuale. Trasmettere la geometria di una o più intersezioni con un singolo messaggio.
- **SPAT (Signal Phase and Time)**: un componente invia un messaggio, chiamato per l'appunto SPAT, con lo stato corrente di ogni fase attiva del semaforo e quanto tempo manca alla prossima fase.

- **HMI (Human Machine Interface)**: questo componente ha la responsabilità di comunicare all'utente informazioni quali avvertenze, pericoli, ecc. (ad esempio mostrandole sul display del cruscotto del veicolo). Come suggerisce il nome, si occupa di far visualizzare al guidatore tutte le informazioni che riceve dagli altri componenti del Framework. Implementa dei meccanismi per decidere quali messaggi sono più importanti da rendere visibili nel quadro di bordo.

I servizi specifici del mondo **WAVE** sono:

- **BSM (Basic Safety Message)**: un componente che gestisce un tipo di messaggio che viene periodicamente inviato dai veicoli. Il messaggio BSM contiene informazioni sulla posizione, la velocità, la dinamica del veicolo. È un messaggio che viene inviato ad alta frequenza, ogni 100 ms circa.
- **TIM (Traveler Information Message)**: un componente che gestisce i messaggi sul traffico, limiti di velocità, segnali stradali e tanti altri. Tramite opportuni meccanismi può inserire un'area di validità nel quale funzionare.
- **WSA (WAVE Service Advertisement)**: messaggio che rappresenta l'annuncio di un ITS.

I servizi specifici del mondo **ETSI** sono:

- **CAM (Cooperative Aware Message)**: questo messaggio ha le stesse funzionalità del BSM (WAVE). Viene inviato periodicamente dall'ITS.
- **DENM (Decentralized Enviromental Notification Message)**: messaggio che viene utilizzato per avvisare gli utenti di un pericolo in strada.

4.3.3 The Use Cases

L'insieme degli Use Cases (Figura 4.3) fornisce 3 importanti funzionalità applicative.
La `prima` è il **Safety** che riguarda tutti i messaggi di allarme e pericolo che il conducente invia/riceve.
La `seconda` è la **Traffic Efficinency** che serve per la gestione del traffico. I campi di applicazione sono molteplici: dal comunicare una velocità consigliata per far si che il conducente arrivi al semaforo mentre è verde ad una gestione degli incroci.
La `terza` funzionalità riguarda l'**Infotainment and Business** e può fornire le più disparate informazione ai viaggiatori, dalla presenza di strutture come ospedali alla segnalazione di luoghi di interesse turistico come musei o ristoranti.

Il **Connectivity Framework** ha lo scopo di implementare tutti gli Use Cases del V2X tramite moduli software separati. Questa architettura a strati (Figura 4.1) garantisce che i moduli (Figura 4.3) non dipendano dalla tecnologia sottostante.

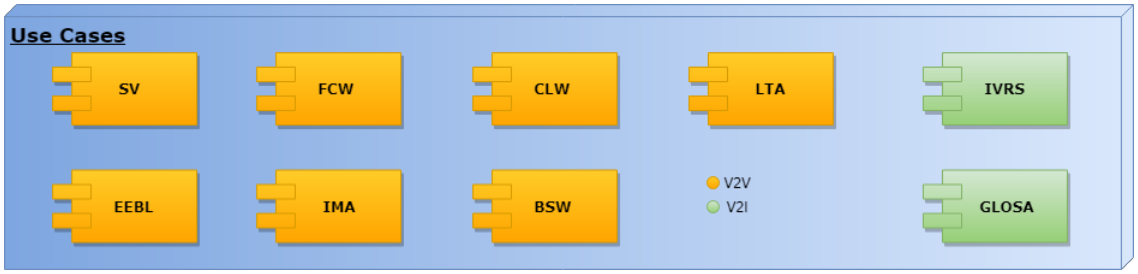

Figura 4.3: Use Cases presenti nel TIC-Framework

Gli **Use Cases** implementati nel **TIC-Framework** sviluppato da Magneti Marelli sono:

- **Stationary Vehicle (SV)**: informa il conducente della presenza di un veicolo fermo sulla stessa strada con le luci di emergenza accese.
- **Forward Collision Warning (FCW)**: avverte il conducente di una possibile collisione imminente con un altro veicolo davanti al traffico.
- **Electronic Emergency Brake Light (EEBL)**: trasmette un messaggio agli altri veicoli dicendo che sta frenando, in modo che chi riceve il messaggio possa evitare la collisione.
- **Green Light Optimised Speed Advise (GLOSA)**: il veicolo suggerisce al conducente una velocità in base al tempo necessario al semaforo per dare o mantenere la luce verde.
- **Intersection Movement Assist (IMA)**: informa il conducente quando è pericoloso entrare in un incrocio a causa di un'alta probabilità di collisione con altri veicoli.
- **Control Loss Warning (CLW)**: avverte tutti altri conducenti che un veicolo è fuori dal controllo del conducente.
- **Left Turn Assist (LTA)**: invia agli altri conducenti durante un tentativo di svolta a sinistra non sicuro. Questa funzionalità si attiva quando c'è un'auto che si avvicina sullo stesso percorso senza intenzione di fermarsi.
- **Blind Spot Warning (BSW)**: informa il conducente che un altro veicolo si trova in un'area cieca.
- **In-Vehicle Road Sign (IVRS)**: Il conducente può visualizzare sul quadro dell'auto informazioni su segnali stradali come limiti di velocità, avvertenze, ecc.

4.4 Hardware Magneti Marelli

Il **Connectivity Framework Magneti Marelli** è stato implementato su diverse piattaforme e architetture, ma è principalmente indirizzato all'hardware embedded basato su Processori **ARM**. I componenti hardware principali che devono essere disponibili a bordo per consentire il corretto funzionamento delle unità software del Framework sono i seguenti:

- **GPS/GLONASS** per ottenere informazioni di interesse geografico (latitudine, longitudine, direzione).
- **802.11p** per abilitare le comunicazioni V2X standard usando i protocolli WAVE ed ETSI-G5.
- **4G/5G** modem per poter funzionare con lo standard 3GPP per il mondo C-V2X.
- **CAN Network** per elaborare i dati del veicolo e per inviarli al display dell'auto.

Diverse board sono state utilizzate attraverso le fasi di sviluppo del Framework, ma in questo lavoro vengono considerate solo la scheda proprietaria **Step3** e un **Car-PC** industriale.

4.4.1 Il Car-PC

Un altro dispositivo su cui è distribuito il Connectivity Framework è il Car-PC. I Car-PC sono computer industriali senza ventola adatti per l'uso all'interno del veicolo. Sono progettati per essere usati per applicazione sia lato veicolo (V2V) che lato infrastruttura (V2I). Quello utilizzato dal gruppo di Technology Innovation è un dispositivo **x86** con le seguenti specifiche:

- Processore Intel Core i7-6820EQ
- 32 GB DDR3 RAM
- 512 GB SSD (espandibili)
- Wi-Fi
- Modem 4G
- Prese VGA, DVI e RCA

4.4.2 Step3 Magneti Marelli

La Board Step3 (Figura 4.4), è una scheda personalizzata Magneti Marelli, sviluppata nella divisione di ElectronicSystems. Si basa sul progetto `Smart Application NXP-Freescale Blueprint forRapid Engineering` (**SABRE**) **i.MX 6QuadPlus** ed è stato sviluppato originariamente per scopi di Infotainment dei veicoli.

Figura 4.4: Foto Esempio della Step3 utlizzata dal gruppo Technology Innovation di Magneti Marelli

La Scheda è stata modificata per le funzioni **V2X** e ha le seguenti specifiche:

- Processore Arm Cortex-A9-based i.MX 6QuadPlus
- 1 GB DDR3L RAM
- 512 MB NAND
- 32 GB eMMC
- CAN High Speed & Low Speed
- GPS, Wi-Fi, Bluetooth
- 4G modem Telit-LE920-EU
- Antenna 802.11p

Per una limitazione del System-on-Chip i.MX 6QuadPlus, la Step3 supporta solo due bus **FlexCAN (Flexible Controller Area Network)**. I moduli FlexCAN forniscono una completa implementazione delle specifiche del protocollo CAN. Per comunicare con gli altri bus CAN presenti sul veicolo, viene utilizzata l'interfaccia PCAN-USB del sistema PEAK.

4.4.3 Modem 4G

Il modem utilizzato dalla **Step3** Magneti Marelli è il `Telit LE-920-EU` con le seguenti caratteristiche tecniche [16]:

- Quad Band GPRS ed EDGE classe 10
- 34 x 40 mm LGA footprint
- Ricevitore GPS / Glonass incorporato
- Conforme con 3GPP versione 9
- Embedded TCP/IP stack
- Set di comandi AT standard ed esteso
- Uplink fino a 50 Mbps
- Downlink fino a 100 Mbps

Il modem include la possibilità di essere utilizzato tramite Comandi AT, che sono stati alla base dello sviluppo della mia Tesi. Tramite questi è infatti possibile la configurazione, il settaggio o il cambiamento di determinate funzionalità del modem al fine di ottenere varie informazioni, tra le quali, ad esempio, lo stato della cella agganciata, la presenza di tutte le celle nel range di visibilità dell'antenna o anche solo parametri riguardanti la qualità e forza del segnale (`RSSI, RSRQ, SNR` etc...).

4.5 Software utilizzati in Magneti Marelli

Il Magneti Marelli Connectivity Framework è completamente scritto nel linguaggio di programmazione **C++**.
Richiede alcune librerie (e driver) per l'interfaccia con le periferiche di bordo come la Rete CAN, il GPS/GLONASS, i dispositivi di connettività V2X e il Modem 4G. La configurazione del Modem è stata fatta tramite il demone **PPPD** mentre sono stati utilizzati i **Comandi AT** per effettuare l'Analisi richiesta dal lavoro di Tesi.

Il processo di compilazione è gestito con **CMake** e tutto il codice sorgente del software viene caricato su un repository **SVN**.

4.5.1 SVN

Il sistema di controllo per lo sviluppo del Connectivity Framework è **SVN (Apache Subversion)**. Ogni componente del Framework è contenuto in un modulo sul repository che è organizzato nella struttura classica di directory SVN:

- `trunk`: questa è la principale area di sviluppo. La versione stabile e definitiva del modulo software dovrebbe essere contenuta qui.

- `branch`: ogni volta che il codice contenuto nel `trunk` ha bisogno di una correzione di bug o di una modifica importante, si crea un `branch`. In questo modo l'integrità del `trunk` può essere preservata e tutte le modifiche vengono fatte su una porzione di repository che non viene inclusa nell'albero di compilazione. Quando un branch viene testato e ritenuto stabile e funzionante, il contenuto viene spostato in `trunk`.

- `tag`: rappresenta la base del software.Ogni volta che i `tag` contengono versioni stabili del modulo software che devono essere preservate, queste vengono spostate qui per diventare il nuovo punto di partenza per gli sviluppi futuri.

4.5.2 Cross-Compilazione

Ogni modulo software viene creato utilizzando **CMake**e **Make**. In ogni unità (`branch/tag/trunk`) è definito un file `CMakeLists.txt` che contiene le direttive di CMake per generare un **Makefile** e compilare correttamente il componente e determinare il grafo delle dipendenze per un particolare output.

Tramite la variabile `CMAKE-TOOLCHAIN-FILE`, lo sviluppatore può specificare un particolare file che permette quella che viene chiamata **Cross-Compilazione**. Tramite la Toolchain infatti viene definito il compilatore da utilizzare per il modulo, l'architettura di destinazione, i flag di compilazione e la directory in cui i binari devono essere inseriti.

Per ogni compilatore per C++ presente nel progetto **TIC-Framework** esiste una versione WAVE ed ETSI in base al tipo di standard che si vuole utilizzare, ed una versione per **ARM** e **x86** in base a se deve essere compilata per la **Step3** o per il **Car-PC**.

4.5.3 KDevelop

Tutto la parte relativa alla **LibGSM**, Cap. 6, è stata sviluppata utilizzato il programma KDevelop.
KDevelop è un software gratuito **IDE (integrated development environment)**, cioè un ambiente di sviluppo integrato, open-source disponibile per Linux, Microsoft e macOS [17].

Generalmente un IDE aiuta lo sviluppatore segnalando possibili errori di sintassi del codice direttamente durante la fase di scrittura, inoltre possiede tutta una serie di tool e funzionalità di supporto alla fase di sviluppo e debugging.

La versione utilizzata è stata la 4.0 che permette l'utilizzo di questo IDE per sviluppare in **C, C++, C# e Java**. Tra le varie funzionalità possiede un supporto per **Doxygen**, il controllo **Subversion** e **Git**, completamento automatico per C e C++ e diversi tipi di gestore del progetto per ogni linguaggio usato, come **automake**, **qmake** per le Qt e **Apache Ant** per il Java.

4.5.4 Sviluppo

In questo ambiente e con queste linee guida ho iniziato a sviluppare gli algoritmi, gli applicativi e i servizi necessari, prima per permettere la comunicazione tra le schede tramite Rete cellulare, poi creando una libreria di supporto con funzioni dedicate per la gestione del Modem e l'analisi della connettività.

Il mio lavoro è stato quindi inserito all'interno del progetto **TIC-Framework**, in questo modo sono riuscito ad interfacciarmi con i servizi offerti dai vari moduli per riuscire ad effettuare la parte di **Testing**, che si è concretizzata nella realizzazione di una serie di file di `LOG`. Effettuando una fase di **Post-Processing** su questi file, sono riuscito a fare un'analisi completa di tutte le informazioni che il Team di sviluppo ha richiesto durante le varie fasi del progetto.

Capitolo 5

Sviluppo Configurazione di Rete

5.1 Introduzione

Una parte del mio lavoro di Tesi è stata quella di configurare la **Board Step3**, in cui era presente una distribuzione Linux personalizzata ad-hoc creata tramite Yocto Project dal team di sviluppo, al fine di ottenere una connessione verso Internet.
A causa delle limitazioni presenti nella board, come in qualunque sistema Embedded, il pool di applicativi installabili ed eseguibili era ovviamente esiguo.

5.2 Alternative Analizzate

In generale le principali alternetive per la connessione tramite modem su linux si basano:

- **Protocolli Open Source**: Protocolli che utilizzano generalemnte gli AT-Command per la Instaurazione/Gestione/Terminazione della connessione. I principali tool esistenti sono: pppd, wvdial, kppp, gnome-ppp. Fanno tutti riferimento al PPP (Point-to-Protocol) e forniscono un'interfaccia **ppp0**.

- **Protocolli Proprietari**: Protocolli proprietari nati dall'esigenza di utilizzare al massimo le velocità offerte dai modem LTE. I principali tool esistenti sono: QMI, QCDM, WMC. Alcuni usano i comandi AT per la Gestione/Inizializzazione della connessione, ma in generale forniscono un interfaccia **wwan0**.

I protocolli proprietari elencati sono in generale più performanti rispetto ai loro competitor Open Source, ma necessitano di componenti driver e software specifici, e un supporto da parte del firmware del Modem.

Uno dei migliori protocolli proprietari, QMI, che fa riferimento alla libreria Libqmi, è stato sviluppato da Qualcomm nel 2013 e solo recentemente sta ricevendo supporto sia da parte dei produttori di Modem per poter essere utilizzato sia da parte della comunità Linux che l'ha inserito nelle ultime versione del Kernel.

Il modem utilizzato in questa fase all'interno del **TIC-Framework** non supporta questa possibilità. Il Team di sviluppo ha infatti optato per un modem temporaneo da usare in questa momento in cui stanno migrando dal mondo Wi-Fi a quello GSM/LTE.
La loro scelta è stata quindi quella di prendere il **Telit-Le920-EU** che soddisfa pienamente le specifiche richieste dal gruppo di lavoro in questo momento permettendo a tutti di iniziare ad effettuare la fase di sviluppo e testing nel mondo GSM, utilizzando i protocolli Open Source, in vista dell'arrivo nel mercato del nuovo Modem sviluppato da Qualcomm per il mondo V2X, denominato **Qualcomm MDM 9150 C-V2X**.

5.3 Sviluppo della Soluzione Proposta

Ho deciso di procedere utilizzando il **PPP (Point-to-Point Protocol)** per creare la connessione verso l'**Internet Service Provider (ISP)** tramite la configurazione del **PPPD (PPP-Daemon)**.

Successivamente ho creato un albero di file di configurazione al fine di permettere una migliore modularità della struttura per far sì che ogni modifica successiva fosse il più semplice possibile. Infine ho sviluppato uno script che permettesse la configurazione automatica tramite l'immissione di pochi parametri necessari per la connessione e modificato alcune regole udev al fine di automatizzare la connettività.

Questa fase si può quindi riassumere in questi step:

- Studio del **PPP** e delle varie alternative di per creare la struttura, Par. 5.4
- Configurazione dei file necessari per la connessione e l'avvio del Demone PPP, Par. 5.6.1
- Sviluppo di due *Script* che automatizzino i processi di avvio della connessione e configurazione dei file necessari, Par. 5.6.3
- Modifica di alcune regole interne al BSP del **TIC-Framework Magneti Marelli** per aggiungere la possibilità di connessione automatica all'avvio della Board, Par. 5.6.4

5.4 Point to Point Protocol (PPP)

Il **PPP (Point-to-Point Protocol)** è un protocollo di comunicazione di livello 2 (nel modello della Pila Protocollare ISO/OSI) utilizzato per stabilire una connessione diretta tra due nodi. Può fornire autenticazione della connessione, crittografia di trasmissione, e compressione [18].

Il PPP viene utilizzato su molti tipi di reti tra cui quelle che usano:

- Cavo Seriale
- Linea Telefonica
- Telefono Cellulare
- Collegamenti Radio Specializzati
- Collegamenti in Fibra Ottica

Gli **Internet Service Provider (ISP)** utilizzano **PPP** per l' accesso remoto del client a Internet, poiché i pacchetti IP non possono essere trasmessi su una linea modem senza alcun protocollo di collegamento dati. In particolare i principali protocolli utilizzati dagli ISP sono **PPPoE (Point-to-Point Protocol over Ethernet)** e **PPPoA (Point-to-Point Protocol over ATM)**.

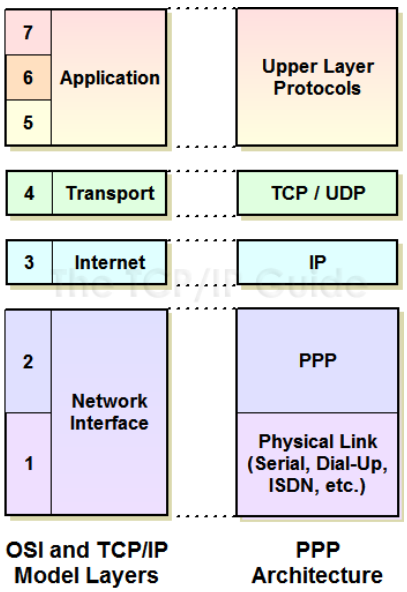

Figura 5.1: PPP funge da interfaccia tra il livello Internet e il livello fisico. Questo corrisponde al livello due nel modello di riferimento OSI [19]

Il **PPP** è stato progettato per funzionare con numerosi protocolli di livello di rete (IP, TRILL, NBF, Apple Talk etc...) e l'unico requisito è che la rete nella quale è utilizzato sia Duplex [20].

PPP è un protocollo a più livelli con tre componenti:

- Un componente di incapsulamento che permette la trasmissione del pacchetto sopra il livello fisico specificato
- Un protocollo **LCP (Link Control Protocol)** per stabilire, configurare e testare il collegamento e negoziare impostazioni, opzioni e l'uso delle funzionalità.
- Uno o più protocolli per il controllo di rete, **Network Control Protocols (NCP)** utilizzati per negoziare parametri di configurazione e strutture opzionali per il livello di rete. Esiste un **NCP** per ogni protocollo di livello superiore supportato da **PPP**.

5.5 PPP Daemon

Il **PPPD (Point-to-Point Protocol Daemon)** è un demone che viene utilizzato per gestire le connessioni di rete tra due nodi su sistemi operativi di tipo Unix. È configurato utilizzando argomenti della riga di comando e/o file di configurazione. Anche se inizialmente è stato utilizzato per gestire solo l' accesso dial-up, adesso è anche utilizzato per gestire connessioni a banda larga [21].

Il ruolo del **PPPD** è quello di inizializzare/gestire/terminare la sessione **PPP**. I suoi compiti sono:

- `Rilevamento di collegamenti in Loop`: tramite l'utilizzo di **Magic Numbers**. Il PPPD quando manda un messaggio LCP include un Magic Number nel pacchetto per capire se sta ricevendo una sua trama o una trama del peer, confrontando i Magic Number con il suo.
- `Autoconfigurazione Automatica`: tramite il protocollo **LCP (Link Control Protocol)** negozia caratteristiche come l'**ACFC (Address-and-Control-Field-Compression)**, caratteri di escape e compressione, crittografia e metodi di autenticazione da utilizzare
- `Autenticazione e Controllo degli Accessi`: tramite i protocolli **CHAP (Challenge-Handshake Authentication Protocol)** e **PAP (Password Authentication Protocol)**

- `Configurazione Livello di Rete`: tramite **IPCP (Internet Protocol Control Protocol)** negozia parametri come l'indirizzo IP, gli indirizzi dei nameserver e l'**MTU (Max Transmission Unit)**
- `Configurazione delle interfacce di rete e delle route`: al termine della fase di negoziazione il demone si occupa di configurare le interfacce di rete con l'indirizzo IP che ha ricevuto e di impostare le route per permettere al sistema di andare su Internet

5.6 Sviluppo della Configurazione

In questo Paragrafo elencherò i vari file di configurazione che ho sviluppato e le modifiche che ho apportato all'interno del sistema.

5.6.1 File di Configurazione

Di seguito è riportato l'albero dei file creati per la configurazione del **PPPD** installati nella Step3.

```
/
└── etc
    ├── ppp
    │   ├── peers
    │   │   └── provider   ⟸
    │   └── chatscripts
    │       ├── connect_chat   ⟸
    │       ├── pin.CODE
    │       ├── pin.NONE
    │       ├── pin
    │       ├── apn.tim
    │       ├── apn.vodafone
    │       ├── apn
    │       ├── mode.ALL
    │       ├── mode.4Gonly
    │       └── mode
```

I file principali di questa struttura sono quelli evidenziati: `/etc/ppp/peers/provider` e `/etc/chatscripts/connect_chat`.

Di seguito adesso il contenuto di questi file, con a lato i commenti per i contenuti più interessanti.

File /etc/ppp/peers/provider:

```
debug           ####Debug info from pppd
connect "/usr/sbin/chat -v -f /etc/chatscripts/connect_chat"
/dev/ttyUSB3    ####Serial Device to which the phone is connected
115200          ####Serial port line speed
dump
noauth          ####The phone is not required to authenticate
user ****
name ****
password ****   #### * stay for privacy
defaultroute
noipdefault     ####pppd must not propose any IP address to the peer
ipcp-accept-local
ipcp-accept-remote
persist         ####Keep modem up even if connection fails
holdoff 0
crtscts         ####Hardware flow control
usepeerdns      ####Ask the peer for up to 2 DNS server addresses
novj
nobsdcomp
novjccomp
nopcomp
noaccomp
nodeflate       ####No ppp compression
lock            ####For sanity, keep a lock on the serial line
show-password   ####Show password in debug messages
```

File /etc/chatscripts/connect_chat:

```
#!/bin/sh
ABORT "NO CARRIER"
ABORT "BUSY"
'' 'ATQ0'
'OK-AT-OK' 'ATZ'
TIMEOUT 3
'OK' @/etc/chatscripts/pin
'OK\d-AT-OK''ATI'
'OK' 'ATZ'
'OK' 'ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0'
'OK' @/etc/chatscripts/mode
'OK' @/etc/chatscripts/apn
'OK' 'ATD*99***1#'
CONNECT ''
```

5.6.2 Instaurazione della connessione

Di seguito gli Output che confermano l'avvenuta instaurazione della connessione a seguito di tutta la configurazione precedentemente elencata.
Rispettivamente sono i messaggi inviati nel file di log durante l'esecuzione del PPPd nel file `/var/log/messages` e il risultato ottenuto dal comando `ifconfig` che permette di verificare che l'interfaccia di rete è stata correttamente creata.

```
daemon.debug pppd[1392]: Script /usr/sbin/chat -v -f /etc/chatscripts/connect_chat finished (pid 1393), status = 0x0
daemon.info pppd[1392]: Serial connection established.
daemon.debug pppd[1392]: using channel 55
daemon.info pppd[1392]: Using interface ppp0
daemon.notice pppd[1392]: Connect: ppp0 <--> /dev/ttyUSB3
daemon.debug pppd[1392]: sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x3f8c7e2c>]
daemon.debug pppd[1392]: rcvd [LCP ConfReq id=0x51 <asyncmap 0x0> <auth chap MD5> <magic 0xeda8478a> <pcomp> <accomp>]
daemon.debug pppd[1392]: sent [LCP ConfRej id=0x51 <pcomp> <accomp>]
daemon.debug pppd[1392]: rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x3f8c7e2c>]
daemon.debug pppd[1392]: rcvd [LCP ConfReq id=0x52 <asyncmap 0x0> <auth chap MD5> <magic 0xeda8478a>]
daemon.debug pppd[1392]: sent [LCP ConfAck id=0x52 <asyncmap 0x0> <auth chap MD5> <magic 0xeda8478a>]
daemon.debug pppd[1392]: rcvd [LCP DiscReq id=0x53 magic=0xeda8478a]
daemon.debug pppd[1392]: rcvd [CHAP Challenge id=0x1 <52ab93596ec2d4edbe3d292b149b94a0>, name = "UMTS_CHAP_SRVR"]
daemon.debug pppd[1392]: sent [CHAP Response id=0x1 <149936cd118f8a5af7617a50d39f29db>, name = "user48@trial5G-4G"]
daemon.debug pppd[1392]: rcvd [CHAP Success id=0x1 ""]
daemon.info pppd[1392]: CHAP authentication succeeded
daemon.notice pppd[1392]: CHAP authentication succeeded
daemon.debug pppd[1392]: sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
daemon.debug pppd[1392]: rcvd [IPCP ConfNak id=0x1 <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: sent [IPCP ConfReq id=0x2 <addr 0.0.0.0> <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: rcvd [IPCP ConfNak id=0x2 <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: sent [IPCP ConfReq id=0x3 <addr 0.0.0.0> <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: rcvd [IPCP ConfNak id=0x3 <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: sent [IPCP ConfReq id=0x4 <addr 0.0.0.0> <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: rcvd [IPCP ConfNak id=0x4 <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: sent [IPCP ConfReq id=0x5 <addr 0.0.0.0> <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: rcvd [IPCP ConfReq id=0x0]
daemon.debug pppd[1392]: sent [IPCP ConfNak id=0x0 <addr 0.0.0.0>]
daemon.debug pppd[1392]: rcvd [IPCP ConfRej id=0x5 <ms-dns1 10.11.12.13> <ms-dns2 10.11.12.14> <ms-wins 10.11.12.13> <ms-wins 10.11.12.14>]
daemon.debug pppd[1392]: sent [IPCP ConfReq id=0x6 <addr 0.0.0.0>]
daemon.debug pppd[1392]: rcvd [IPCP ConfReq id=0x1]
daemon.debug pppd[1392]: sent [IPCP ConfAck id=0x1]
daemon.debug pppd[1392]: rcvd [IPCP ConfNak id=0x6 <addr [redacted]>]
daemon.debug pppd[1392]: sent [IPCP ConfReq id=0x7 <addr [redacted]>]
daemon.debug pppd[1392]: rcvd [IPCP ConfAck id=0x7 <addr [redacted]>]
daemon.warn pppd[1392]: Could not determine remote IP address: defaulting to 10.64.64.64
daemon.notice pppd[1392]: local  IP address 91.81.200.83
daemon.notice pppd[1392]: remote IP address [redacted]
daemon.debug pppd[1392]: Script /etc/ppp/ip-up started (pid 1402)
daemon.debug pppd[1392]: Script /etc/ppp/ip-up finished (pid 1402), status = 0x0
```

Figura 5.2: Output del PPP Daemon dopo l'instaurazione della connessione

```
ppp0      Link encap:Point-to-Point Protocol
          inet addr:[redacted]  P-t-P:10.64.64.64  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:1028 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1114 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:525857 (513.5 KiB)  TX bytes:532661 (520.1 KiB)
```

Figura 5.3: Output del comando Linux ifconfig

5.6.3 Creazione degli Script

Nella fase successiva alla creazione e configurazione dei file utili per il **PPPD** ho proceduto a sviluppare due script:

1. Il primo, chiamato **startPPPconnection.sh**, è stato sviluppato al fine di automatizzare il lancio del demone all'avvio della Board, per ottenere fin da subito connettività contestualmente all'avvio del Framework. Questo Script è stato inserito all'interno del "pool" di script già presenti nel **TIC-Framework**, ed è stato quindi opportunamente programmato per adattarsi alle esigenze del progetto **V2X**.

2. Il secondo, chiamato **configuration.sh**, è stato sviluppato al fine di facilitare la creazione e/o la modifica dei file di configurazione da parte di un qualunque sviluppatore interno al progetto. Tramite questo Script ho implementato la possibilità di modificare solo i file necessari nei casi di cambio SIM o simili. Per funzionare a prescindere dal pacchetto di sviluppo che il programmatore sta testando in quel momento; ovviamente sono state fatte delle modifiche durante la creazione del **BSP** della **Step3** per avere un background di partenza comune a cui fare riferimento.

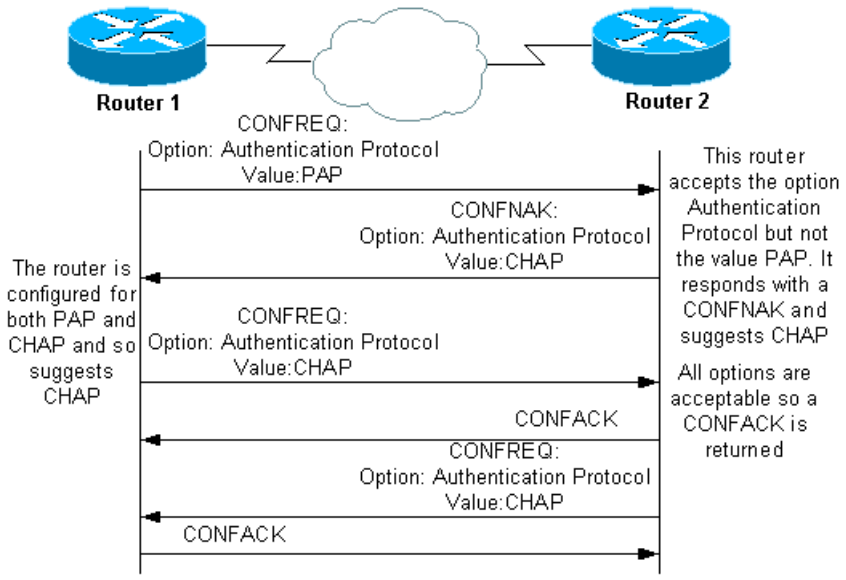

Figura 5.4: Esempio di scambio di messaggi nel PPP per l'instaurazione della connessione [22]

5.6.4 Modifica al BSP della Step3

Il **BSP (Board Support Package)** è il livello software contenente driver specifici dell'hardware e altre routine che consentono a un particolare sistema operativo (tipicamente un sistema operativo in Real-Time) di funzionare in un particolare ambiente hardware.

Un **BSP** è personalizzabile, consentendo all'utente di specificare quali driver e routine devono essere inclusi nella build in base alla loro selezione di opzioni hardware e software.
Il **BSP** del team di sviluppo per il **TIC-Framework** è stato quindi modificato per supportare le loro esigenze eliminando tutto il superfluo.

La parte finale di questo lavoro è stata quindi quella di apportare le dovute modifiche al **BSP** per far sì che ogni **Step3** utilizzata dal team di lavoro possedesse tutte quelle caratteristiche software necessarie per inizializzare/gestire/terminare/configurare la connessione, come ho abbondantemente specificato nei paragrafi precedenti.

5.7 Infrastruttura di Rete

Una volta configurata la **Board Step3** per ottenere la connettività, il lavoro successivo è stato quello di scegliere un'infrastruttura di rete che supportasse al meglio tutte le funzionalità del **Framework**.

A causa del cambio di tecnologia che ha subito il progetto **TIC-Framework**, passando da Wi-Fi 802.11p al 4G/5G, è stato necessario cambiare anche il livello dell'infrastruttura di rete.

Sono state valutate diverse opzioni per la versione **Long-Range V2X**, che a causa della mancanza dell'interfaccia PC5 per il Broadcast, necessita di un'alternativa per permettere la comunicazione V2V tramite una topologia a centro stella.

Le proposte analizzate sono state tre:

- Creazione di una **VPN**
- Utilizzi di servizi di **Dynamic DNS**
- Utilizzo di indirizzi **IP Pubblici Statici**

5.7.1 Proposta N°1: VPN

La prima possibilità offerta è quindi quella riguardante la creazione di una VPN in cui i veicoli dopo aver ottenuto l'accesso ad internet, comunicheranno tramite l'IP offerto loro dal server-VPN. Queste tecnologie sono standardizzate nella Internet Engineering Task Force (IETF) [23].

Creando questa infrastruttura si otterrebbero i seguenti **Vantaggi**:

1. Gestione degli indirizzi IP: questi sarebbero allocati staticamente e in maniera esclusiva per ogni veicolo. In questo modo il centro stella dovrebbe solo ridondare ogni messaggio ricevuto su una lista di indirizzi abilitati sulla stessa VPN.

2. Unire centro stella e server-VPN: concentrare all'interno di un'unica entità sia la figura del centro stella che del gestore della VPN, si eviterebbe di ridondare l'informazione troppe volte tra i due nodi focali di questa configurazione.

3. Sicurezza: Possibilità di aggiungere un livello di crittografia e autenticazione dei messaggi scambiati passando ad una Secure-VPN, che mantiene l'infrastruttura della precedente aggiungendo la sicurezza necessaria.

Questa proposta non è risultata la migliore a causa dei seguenti **Svantaggi**:

1. Inserimento di un ritardo: causato dall'incapsulamento di ogni unità informativa all'interno di un pacchetto VPN. Questo ritardo viene aggiunto per ogni messaggio trasmesso. Purtroppo nel progetto TIC-Framework il rate dei pacchetti in uscita è di circa 100 ms, nel momento in cui si aggiungono altri fattori, spesso impredicibili, di ritardo si rischia di compromettere la corretta operatività e funzionalità dello stesso.

2. Impossibilità di unire il centro stella con il server-VPN: a causa della progettazione del Packet Filter, nato a partire dal Framework montato nelle OBU, questi non può essere configurato per gestire sia la parte di Rete relativa alla gestione della VPN che quella relativa al suo funzionamento di Packet Filter. Questo ha aumentato il delay per ogni singolo pacchetto visto che la maggior parte del traffico veniva ridondato tra il centro stella e il server per ottenere le funzionalità richieste.

In un secondo momento, quando l'interfaccia PC5 sarà disponibile, la soluzione tramite VPN per la rete di tipo V2N potrebbe tuttavia essere la migliore per soddisfare molti requisiti (sicurezza, autenticazione etc...) propri di questo tipo di comunicazione veicolare.

5.7.2 Proposta N°2: Dynamic DNS

La seconda possibilità riguarda l'utilizzo di Servizi di DNS Dinamici. Questa tecnologia permette di associare ad un nome **DNS (Domain Name System)** sempre un indirizzo **IP** anche se quest'ultimo cambia nel tempo.

I nomi in un DNS server sono normalmente associati stabilmente ad indirizzi IP statici, i quali a loro volta sono assegnati ad host che hanno funzionalità di server. Molti dispositivi, in particolare quelli che si collegano ad internet utilizzando i servizi di uno (o più) **ISP (Internet Service Provider)**, ricevono invece un indirizzo diverso ad ogni connessione, ed è questo il caso di di chi utilizza una SIM per navigare.

In questo modo è impossibile raggiungerli da internet, perché non si conosce il loro indirizzo IP a priori. Ciò, teoricamente, preclude la possibilità di amministrarli remotamente e di offrire servizi su questi dispositivi.

Il **Dynamic DNS** permette a questi host di essere sempre raggiungibili attraverso il loro nome DNS, previa registrazione, e quindi rende possibile amministrarli remotamente ed erogare servizi raggiungibili da chiunque su internet [24].

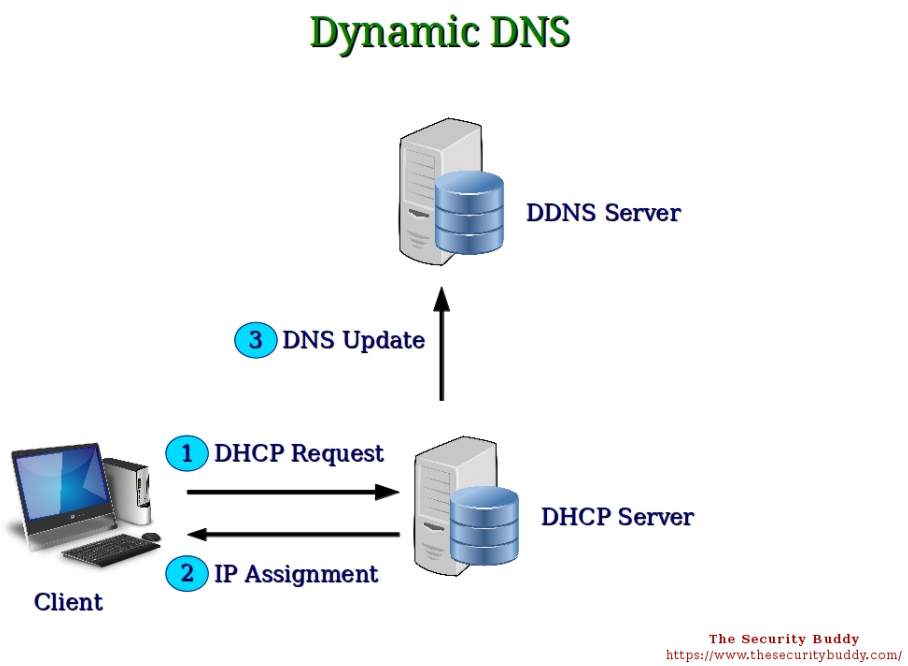

Figura 5.5: Esempio di funzionamento base del Dynamic DNS

5.7.3 Proposta N°3: IP Pubblici Statici

La terza possibilità riguardava l'utilizzo di SIM configurate dall'operatore per ottenere un indirizzo IP Statico, Pubblico e Dedicato ad ogni accesso.
Ogni indirizzo IP infatti può essere:

Statico significa che l'indirizzo IP non cambia mai finché si rimane con lo stesso provider o lo stesso server.

Dinamico significa che l'indirizzo IP può cambiare di volta in volta.

Oltre a questo può avere uno dei seguenti attributi:

Pubblico significa che l'indirizzo IP può essere visitato da qualsiasi computer nel mondo.

Privato significa che l'indirizzo IP può essere utilizzato solo da quelli sulla stessa rete.

Condiviso significa che altre persone usano il tuo indirizzo IP per la loro connessione o siti web.

Dedicato significa che nessun altro usa il tuo indirizzo IP per la loro connessione o siti web.

Scegliendo questa soluzione si avrebbe il totale controllo sullo stato di ogni veicolo che sarebbe raggiungibile da internet sempre allo stesso indirizzo e da qualunque punto del mondo.
Questo aumenta la semplicità di sviluppo per i programmatori all'interno del progetto a patto di porre molta attenzione ai lato di sicurezza e scalabilità.

I **vantaggi** di questa soluzione sono:

- Nessun delay dovuto a servizi DNS
- Riduzione di costi e gestione di servizi VPN
- Raggiungibilità da qualunque punto nel mondo
- Semplicità di gestione e controllo della connessione di ogni veicolo
- Riduzione dei tempi di ri-connessione grazie alla evitabilità di un DHCP server

Invece i principali **svantaggi** di questa soluzione sono:

- Costi mensili con gli ISP per mantenere questa configurazione nel tempo
- Aumento dei rischi di Sicurezza che comporta un miglioramento dei meccanismi di controllo di ogni connessione in entrata ed uscita
- Poca scalabilità della soluzione

5.7.4 Soluzione Finale

Alla fine la soluzione che è stata ritenuta migliore valutando costi di attivazione, gestione, utilità, sicurezza e implementazione è stata la proposta di utilizzare degli indirizzi IP Pubblici e Statici (Par. 5.7.3), ognuno dedicato ad ogni Board montata su ogni veicolo.

Le motivazioni dietro questa scelta sono molteplici. Nonostante infatti un servizio di VPN (Par. 5.7.1) sembrava essere quello che più di ogni altro riusciva ad equilibrare i costi di gestione e instaurazione con un buon livello di sicurezza, all'interno del progetto per il **Long-Range V2X** questa è stata vista troppo inefficiente e limitante per le caratteristiche architetturali del **TIC-Framework**.

Come già discusso in precedenza, infatti, un modello di infrastruttura di rete per il Long-Range V2X è stato valutato sulla base di parametri insoliti rispetto alle classiche reti IoT.

A tal proposito la soluzione con IP statici e pubblici è stata scelta come la migliore proprio per la sua semplicità di gestione e per i tempi brevi per cui questa fase intermedia del progetto sarà interessata.
Probabilmente, infatti, una volta montato un Modem con interfaccia PC5 e ovviato al problema del V2V che non sarà più simulato, l'infrastruttura di rete verrà nuovamente riconsiderata e rimodellata per venire incontro alle nuove esigenze del progetto.

Capitolo 6

Sviluppo della Libreria LibGSM

6.1 Introduzione

La seconda parte del mio lavoro è stata quella di creare una Libreria in C++ da inserire nel Framework che fornisca delle API utili per la gestione della connessione **LTE**, in generale di qualunque connessione **GSM (Global System for Mobile Communications)**. Per questo motivo è stata chiamata **LibGSM** e fornisce agli applicativi che la utilizzano una serie di funzioni che permettono di ottenere le informazioni riguardanti ad esempio lo stato delle rete, lo stato della connessione o lo stato della Sim.

L'obiettivo di questa parte della mia Tesi è stato quindi quello di creare un middlelayer tra gli applicativi del Framework che necessitavano: o di ottenere informazioni specifiche e puntuali riguardo determinati aspetti della connessione o di configurare determinati parametri riguardanti la gestione del modem o della connessione dati.

Questo libreria quindi è stata sviluppata per fornire sempre le stesse API agli applicativi di livello superiore a prescindere da cambiamenti che possono avvenire a livello fisico, come sostituire un modem con un altro.
In generale per comunicare con un modem l'unico modo per farlo è attraverso gli **AT Commands**, che sono delle stringhe inviate per produrre operazioni quali, ad esempio, composizione, riaggancio e modifica di parametri della connessione.

Il set di comandi è costituito da una serie di stringhe di testo brevi che possono preparare il modem per la comunicazione, impostando caratteristiche come il tipo di connessione, i tempi di attesa o la rilevazione del segnale di occupato. La stragrande maggioranza dei modem dial-up usa i comand AT impostati in numerose varianti in base al produttore.

6.2 AT Commands

Gli **AT Commands**, conosciuti anche come **Hayes Command Set**, sono uno specifico set di comandi originalmente sviluppato da **Dennis Hayes** per il **modem Hayes Smartmodem da 300 baud**, apparecchio costruito dalla sua azienda, la **Hayes Microcomputer Products** [25].

Ogni funzionalità del modem è gestita dal relativo Comando AT. Per inviare un comando occorre trasmettere sulla porta seriale del modem una stringa formata dalle lettere **AT** seguite dal comando e terminata dai carattere di ritorno a capo **CR+LF**.

Esistono due tipi di modalità in un modem, definite nel set di comandi AT:

- **Data Mode**: modalità in cui il modem considera tutto ciò che riceve dal computer come dati e lo invia attraverso la linea telefonica. Quando il modem si trova in questa modalità tutti i caratteri inviati al modem devono essere trasmessi all'host remoto. Il modem entra in modalità dati immediatamente dopo aver effettuato una connessione. Ad esempio, se **ATDT123456789** ha dato luogo a una chiamata telefonica a cui ha risposto un altro modem, il modem segnala la parola **"CONNECT"** e quindi passa alla modalità dati. Qualsiasi altro carattere ricevuto tramite il collegamento seriale viene considerato proveniente dall'host remoto e qualsiasi carattere inviato viene trasmesso allo stesso host.

- **Command Mode**: modalità in cui il modem considera qualsiasi carattere inviato ad esso come comandi locali da eseguire, secondo lo standard **Hayes Command Set**. Un comando è preceduto dalle lettere **"AT"**, che rappresentano **"Attention"**. Ad esempio, se un modem riceve **"ATDT123456789"** mentre è in modalità di comando, lo interpreta come un'istruzione per comporre i numeri **123456789** sul telefono, utilizzando la composizione a toni. In questa modalità, il modem invia le risposte al computer con l'esito del comando. Ad esempio, il modem potrebbe rispondere con la parola **"BUSY"** (occupato) in risposta al comando **ATDT** (comando per effettuare una chiamata telefonica), se sente un segnale di occupato dopo la composizione, ed è stato configurato per ascoltare i segnali di occupato.

Originariamente infatti il set di comandi copriva solo quelle operazioni supportate dai primi modem a 300 bit/s. Quando sono stati necessari necessari nuovi comandi per controllare funzionalità aggiuntive nei modem a velocità più elevata, una varietà di standard una tantum emerse da ciascuno dei principali fornitori. Questi continuarono a condividere la struttura di comando e la sintassi di base, ma aggiunsero dei nuovi comandi usando una sorta di carattere di prefisso, ad esempio '&' per Hayes o '\' per Microcom [26].

I comandi AT si possono suddividere in 4 categorie:

- **AT Commands di base**: Un carattere maiuscolo seguito da una cifra. Ad esempio, Q0.
- **AT Commands estesi**: Un "&" (e commerciale) e un carattere maiuscolo seguito da una cifra. Questo estende il set di comandi di base.
- **AT Commands proprietari**: In genere si inizia con una barra rovesciata ("\") o con un segno di percentuale ("%"); questi comandi variano ampiamente tra i produttori di modem.
- **AT Commands di registro**: Scritti nella forma Sr = n dove r è il numero del registro da modificare, e n è il nuovo valore assegnato.

Nel 1999 è stato definito dal 3GPP TS 27.007 Rel. 1 (nel 2018 è stata definita la Rel. 15 [27]) un nuovo standard per uniformare i produttori di modem GSM per offrire tutti le funzionalità base.
I modem GSM/3G/4G in genere supportano le estensioni del set di comandi AT 3GPP TS 27.007, anche se il numero di comandi implementati varia. La maggior parte dei fornitori di modem USB, come Telit, Huawei, Sierra Wireless, ha anche definito estensioni proprietarie per la selezione della modalità radio (preferenza 2G/3G/4G) o simile. Il set-up richiede estensioni di AT Commands specifiche del fornitore, che a volte sono disponibili, altre volte il fornitore richiede una **NDA (Non Disclosure Agreement)** per l'accesso a queste.

Comando	**Descrizione**
`AT&V`	Ritorna la configurazione corrente del modem
`AT+CPIN=1234`	Inserisce il codice PIN 1234
`AT+CSQ`	Ritorna la potenza del segnale nel formato: `+CSQ:<rssi,ber>`
`AT+COPS=?`	Ritorna la Lista di tutte le rete disponibili in quella zona aggiungendo un numero da 0 a 3 per ogni rete per identificarli: 0 - Unknown/ 1- Active/ 2 - Current/ 3 - Forbidden
`ATD*99#`	Compone il numero dell'Access Point
`AT+CGDCONT?`	Ritorna i dati del PDP context necessario per l'instaurazione della connessione
`AT+CIND?`	Ritorna un vettore di interi ognuno rappresentate un diverso tipo di informazione. ad esempio lo stato di roaming, qualità del servizio o il livello di batteria
`AT+CGREG?`	Ritorna Informazioni sullo Stato di registrazione della Sim, notificando in alcuni casi anche CID e LAC della cella servente

Tabella 6.1: Alcuni esempi dei comandi standardizzati dal 3GPP per i modem GSM

6.3 LibGSM

Lo sviluppo della **LibGSM** è stato condotto seguendo 3 fasi:

- **Fase 1 - Lista dei requisiti**: insieme al team di sviluppo è stata stilata una lista delle funzionalità necessarie e prioritarie per il **Framework** e per i servizi che era necessario offrire al gruppo di lavoro per effettuare al meglio i loro Test.
- **Fase 2 - Sviluppo dell'Architettura Software e delle API**: questa fase è stato il fulcro del mio lavoro di Tesi, dopo aver discusso dell'architettura da seguire per lo sviluppo della Libreria, mi sono concentrato nella ricerca degli **AT Command** necessari a svolgere le funzionalità che mi avevano richiesto nella **Fase 1** e nello sviluppare in **C++** le API attraverso l'utilizzo di questi comandi.
- **Fase 3 - Integrazione con il Framework e Testing delle funzioni**: nella fase finale del mio lavoro ho integrato il mio componente all'interno del progetto **TIC-Framework** aggiungendo tutto il necessario per un perfetto inserimento. Durante questa fase ho creato anche un applicativo di Test che si iscrivesse come Use Case al fine di testare tutte le funzioni da me sviluppate. La fase di testing è stata fatto sia a banco che su veicolo e verrà discussa in maniera più esaustiva nel Capitolo seguente.

LibGSM è quindi una libreria che sfrutta gli **AT Commands** per ottenere informazioni dal modem (o anche configurare parametri) permettendo agli sviluppatori che la utilizzeranno di non dover imparare ogni comando AT necessario ai loro scopi. Inoltre si occupa anche di decodificare l'informazione ottenuta dal modem in formato **Raw Data**, cioè "dato grezzo" spesso ottenuto semplicemente da una sequenza alfanumerica, passando ad una semantica più fruibile e più semplice da capire.

Per fare un esempio apriamo Microcom, un programma terminale minimalista per l'accesso ai dispositivi, e vediamo che l'output del comando **AT+CIND?** non è altro che una stringa di numeri separati da virgole.

```
AT+CIND?
+CIND: 4,3,1,0,0,0,0,0,2,0,0
+OK
```

L'unico modo quindi per scoprire a cosa fa riferimento ogni valore della stringa è quello di lanciare il comando che permetta al modem di specificare qual è il formato logico con cui ha composto questo output.

In questo caso basterebbe lanciare il comando **AT+CIND=?** per ottenere la lista completa del significato di ogni valore della stringa e il suo range di valori.
Di seguito l'output del modem **Telit-LE920-EU** [28]:

```
AT+CIND=?
+CIND: (‘‘battchg’’,(0-5,99)),(‘‘signal’’,(0-7,99)),(‘‘service’’,(0-1)),\\
(‘‘sounder’’,(0-1)),(‘‘message’’,(0-1)),(‘‘call’’,(0-1)),(‘‘roam’’,(0-1)),\\
(‘‘smsfull’’,(0-1)), (‘‘rssi’’,(0-5,99)),("GPRS coverage",(0-1)),\\
("callsetup",(0-3))
+OK
```

Spesso mi sono avvalso di strutture complesse come HashMap o semplici Mappe al fine di schematizzare e rendere più "User Friendly" i dati ottenuti.
In questo modo l'esempio precedente potrebbe essere contenuto in una Map Key-Value dove la Key sarebbe il nome del parametro e il Value il valore preso dalla stringa del primo comando, ottenendo una struttura del genere:

```
Map = {
"battchg" , "4"
"signal" , "3"
"service" , "1"
"sounder" , "0"
"message" , "0"
"call" , "0"
"roam" , "0"
"smsfull" , "0"
"rssi" , "2"
"GPRS coverage" , "0"
"callsetup" , "0"
}
```

Questo esempio è proprio la descrizione della mia API nella LibGsm che ho chiamato **getMapIndicatorControl** che crea una struttura del genere e la restituisce al chiamante rimodellata come elencato in precedenza.
In questo modo allo sviluppatore basterà accedere al dato nella Map a cui è interessato mentre l'operazione di esecuzione dei comandi AT e traduzione in una logica comprensibile è rilasciata alla mia funzione.

6.3.1 UML Class Diagram

Di seguito i Class Digram UML della mia Libreria.

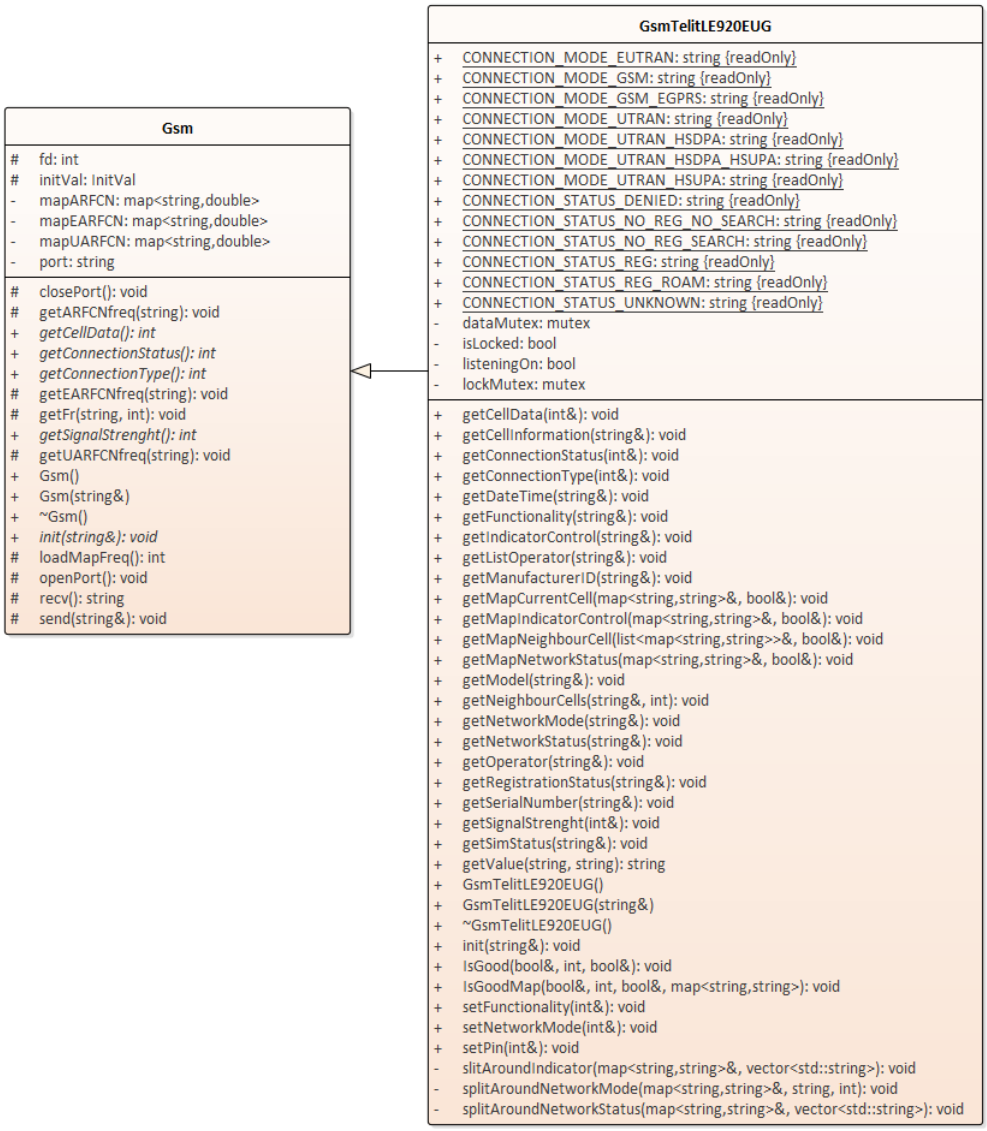

Figura 6.1: UML Class Diagram (prima parte)

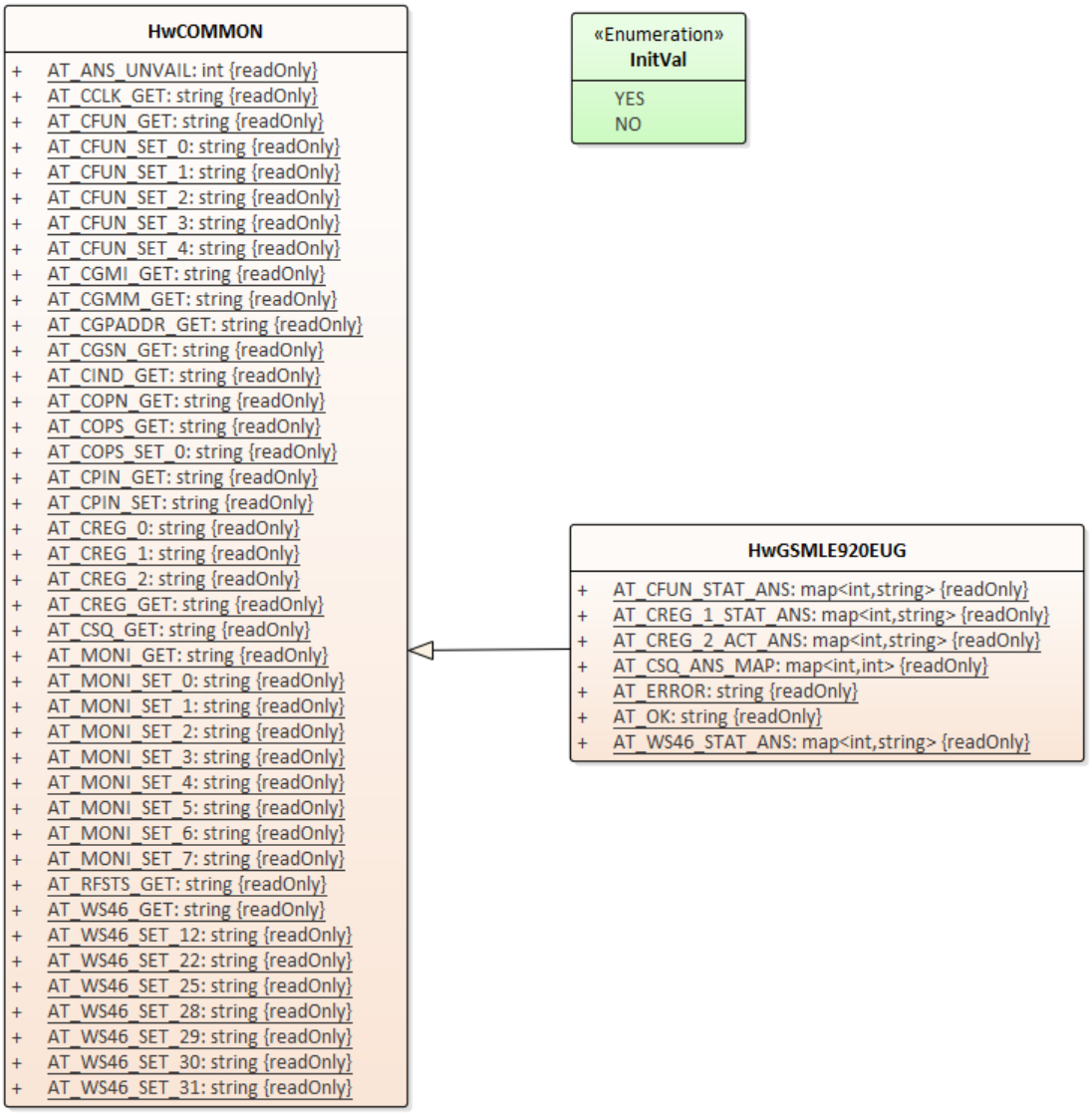

Figura 6.2: UML Class Diagram (seconda parte)

6.4 API Sviluppate

Di seguito la lista (con una brevissima descrizione) con tutt le API sviluppate, testate e attualmente funzionanti.

Lista - **"Initialization"**:

- **GsmTelitLE920EUG**: Costruttore della classe, crea i file e setta i parametri necessari alla comunicazione con il modem tramite comandi AT.
- **~GsmTelitLE920EUG**: Distruttore della classe, rilascia tutte le risorse allocate.
- **init**: Configura la porta seriale del Device nella quale è connesso il modem. Può essere letta da file di configurazione.

Lista API - **"Get Functions"**:

- **getCellInformation**: Restituisce l'output RAW relativo alle informazioni della cella servente in quel momento, tra le varie informazioni sono presenti valori sulla cella stessa come Cell-ID e TAC ma anche dati sulla qualità del segnale e la banda utilizzata.
- **getMapCurrentCell**: Restituisce un Map, basato sul metodo precedente `getCellInformation`, e contiene per ogni riga la coppia "Nome Parametro" e "Valore". Vedi Approfondimento nel Par. 6.5.3.
- **getDateTime**: Restituisce l'ora attuale da internet.
- **getFunctionality**: Restituisce lo stato di TX/RX (Trasmissione e Ricezione) del Modem.
- **getIPaddress**: Restituisce l'indirizzo IP se il device è correttamente connesso alla rete Internet.
- **getListOperator**: Restituisce la lista di tutti gli operatori presenti sulla SIM.
- **getManufacturerID**: Restituisce l'ID identificativo del Modem.
- **getIndicatorControl**: Restituisce l'output RAW del comando AT+CIND? che rappresenta una stringa di numeri ognuno rappresentante un parametro ben definito sullo stato del Modem.
- **getMapIndicatorControl**: Restituisce un Map, basato sul metodo precedente `getIndicatorControl`, e contiene per ogni riga la coppia "Nome Parametro" e "Valore". Vedi Approfondimento nel Par. 6.5.5.

- **getNeighbourCells**: Restituisce l'output RAW relativo alle informazioni delle celle vicine e visibili, alle quali il modem potrebbe connettersi, tra le varie informazioni sono presenti valori sulle celle stessa come Cell-ID e TAC ma anche dati relativi alla qualità del segnale e la banda utilizzata.

- **getMapNeighbourCell**: Restituisce un Map, basato sul metodo precedente `getNeighbourCells`, e contiene per ogni riga la coppia "Nome Parametro" e "Valore". Vedi Approfondimento nel Par. 6.5.4.

- **getNetworkStatus**: Restituisce l'output RAW relativo alle informazioni della cella servente in quel momento, molto simile al `getCellInformation`, utilizza un comando AT differente da quest'ultimo e di conseguenze alcune informazioni presenti in uno dei due non sono presenti nell'altro, per completezza sono stati implementati entrambi.

- **getMapNetworkStatus**: Restituisce un Map, basato sul metodo precedente `getNetworkStatus`, e contiene per ogni riga la coppia "Nome Parametro" e "Valore".

- **getModel**: Restituisce il modello del Modem.

- **getNetworkMode**: Restituisce l'impostazione della SIM relativa alla tipologia di rete consentita, cioè se la configurazione attuale permette lo switch tra differenti tipologie di rete (2G, 3G, 4G). Vedi Approfondimento nel Par. 6.5.6.

- **getOperator**: Restituisce l'operatore telefonico a cui si è connessi.

- **getRegistrationStatus**: Restituisce lo stato di registrazione della SIM, cioè se è agganciato ad una cella servente in maniera corretta.

- **getSerialNumber**: Restituisce il Numero Seriale del modello del Modem.

- **getSignalStrenght**: Restituisce la potenza del segnale direttamente in *dbm*, corrisponde all'RSSI del segnale.

- **getSimStatus**: Restituisce lo stato della Sim (se è attiva, bloccata, in attesa di PIN/PUK etc..).

- **IsGood**: Restituisce una valutazione sulla bontà del segnale prendendo come riferimento vari parametri, al suo interno esegue un'operazione sul metodo `getMapCurrentCell` in maniera trasparente all'utilizzatore. Vedi Approfondimento nel Par. 6.5.8.

- **IsGoodMap**: Restituisce una valutazione sulla bontà del segnale prendendo come riferimento vari parametri, a differenza del comando precedente però vuole in input una Map ottenuta dal metodo `getMapCurrentCell`.

Lista API - **"Set Functions"**:

- **setFunctionality**: Imposta lo stato di TX/RX (Trasmissione e Ricezione) del Modem.

- **setNetworkMode**: Imposta la configurazione della SIM relativa alla tipologia di rete consentita, cioè se l'impostazione attuale permette lo switch tra differenti tipologie di rete (2G, 3G, 4G). Vedi Approfondimento nel Par. 6.5.7.

- **setPin**: Inserisce il codice PIN per sbloccare la SIM (qualora fosse necessario) restituendo anche un valore per confermare l'avvenuta operazione o un possibile errore.

Lista **"Private Functions"**.
Queste funzioni sono state implementate solo per essere fruibili da altre al fine di ottimizzare e migliorare lo sviluppo della Libreria:

- **buffCleanOK**: si occupa di ripulire la Stringa di output di un qualunque comando AT che non abbia segnalato alcun errore.

- **getValue**: funzione di ricerca e ottenimento del valore basato su una keyword all'interno della stringa per gli output di quei comandi che restituiscono una lista di dati nella forma "Nome: Value".

- **slitAroundIndicator**: Metodo utilizzato, da `getMapIndicatorControl`, per effettuare una divisione e riorganizzazione sull'output dell'AT-Command `AT+CIND?`, rilasciato in formato di una stringa di valori numerici separati da virgola. Questo metodo è specifico per il Modem Telit-LE920-EU in quanto ogni costruttore decide quali e quanti valori visualizzare.

- **splitAroundNetworkMode**: Metodo utilizzato, da `getMapCurrentCell`, per effettuare una divisione e riorganizzazione sull'output dell'AT-Command `AT#MONI`, rilasciato in formato di una stringa di valori alfanumerici separati da virgola. Questo metodo è specifico per il Modem Telit-LE920-EU in quanto ogni costruttore decide quali e quanti valori visualizzare.

- **splitAroundNetworkStatus**: Metodo utilizzato, da `getMapNetworkStatus`, per effettuare una divisione e riorganizzazione sull'output dell'AT-Command `AT#RFSTS`, rilasciato in formato di una stringa di valori alfanumerici separati da virgola. Questo metodo è specifico per il Modem Telit-LE920-EU in quanto ogni costruttore decide quali e quanti valori visualizzare.

6.5 Descrizione delle principali API

In questa Sezione descriverò le API che più ho usato durante la fase di Testing, Cap. 7, e che più sono state richieste con maggiore priorità dal Team di Innovation Connectivity di Magneti Marelli.
La struttura di queste API è stata discussa con il Team al fine di essere integrata nel Framework, e durante la fase di Progettazione sono state stilate una serie di vincoli e regole da seguire per procedere nel migliore dei modi.

6.5.1 Linee Guida

Il modello di sviluppo di questi metodi è stato scelto per meglio integrarsi all'architettura del Framework. Per queste ragioni sono state scelte, per lo sviluppo di ogni API, le seguenti linee guida:

- Ogni funzione deve avere il ritorno a **Void**, sia che l'API sia di GET o SET, in questo modo si limita l'allocazione ridondata dello stesso dato, che in caso di strutture complesse può risultare oneroso. Inoltre può essere implementata in futuro una funzionalità basata condition variables su quelle variabili passate come input e poi modificate.

- Le strutture e/o variabili in ingresso alle funzioni devono essere passate sempre per **reference (&)**, rimandando così la gestione dell'allocazione e liberazione delle risorse all'applicativo di livello superiore, snellendo il controllo e la gestione logica di ogni API.

- Si deve progettare la Libreria per essere il più modulare possibile, permettendo in futuro una maggiore scalabilità. In questo modo ogni API sviluppata per un determinato modello di Modem può essere semplicemente sostituita da una nuova implementazione della stessa lasciando però intatto il funzionamento per gli applicativi di livello superiore che utilizzano questi metodi.

- Ogni API deve essere atomica, al suo interno infatti deve eseguire tutte le operazioni necessarie in maniera esclusiva e senza interferenze esterne, per mantenere questa specifica mi sono avvalso di struttura quali i **mutex**, per avere un accesso esclusivo a tutte le risorse necessarie per l'esecuzione di un determinato compito.

6.5.2 Gestione degli Errori

Sulla base delle informazioni ottenute dagli AT Commands e sulla loro sintassi viene creata una struttura apposita decisa in base alle specifiche dell'API, in questo modo si ha un accesso diretto al parametro di riferimento.

Nel caso in cui qualche errore dovesse presentarsi durante l'esecuzione di un comando AT o in qualunque altro momento, sono stati previsti e implementati due meccanismi di supporto:

- Il primo riguarda la gestione dell'**eccezioni** (Exception) che rilancia al chiamante qualunque errore si dovesse verificare. Queste eventualità gestisce i conflitti dovuti a un qualche componente esterno che magari corrompe i dati del modem effettuando chiamate illecite sullo stesso o semplicemente durante uno dei tanti possibili errori generati dal codice.

- Il secondo è di tipo software, è stata implementata una variabile **booleana** (*true* o *false*) relativo alla bontà del dato ottenuto, nel caso infatti il comando AT dia un output differente dal normale, nella sintassi o nella forma, o la composizione della struttura viene compromessa senza rilanciare eccezioni, allora questa variabile fornisce al programmatore la possibilità di controllare quanto è affidabile il dato ottenuto.

Questi meccanismi nascono dalla necessità di avere un controllo sul comportamento dei comandi AT che spesso non esenti da errori di lettura o scrittura impedendo la corretta esecuzione del comando stesso.
Gli stessi comandi infatti richiedono spesso tempi lunghi (qualche secondo) per essere eseguiti.

Questo genere di gestione è stato scelto per far si che il componente gestisca al meglio ogni parte del programma impedendo crash indesiderati che potrebbero in futuro compromettere il corretto funzionamento non solo suo ma anche di tutto il Framework.

In ogni API della LibGSM è sempre presente il primo caso di gestione dell'errore, quello riguardante il rilancio dell'eccezioni, mentre il meccanismo relativo alla variabile booleana è stato implementato solo in quelle API un po' più complesse che utilizzino più di un AT Command o che compongano strutture non semplici come Mappe o Liste.

6.5.3 getMapCurrentCell

Input	***Output***	***AT command(s) used***
map<string, string> & boolean &	Void	AT#MONI=0 AT#MONI

Tabella 6.2: API getMapCurrentCell

Questa API permette di costruire una struttura di tipo `std::map<string, string>` che conterrà al suo interno una riga per ogni coppia Chiave-Valore relativa alle informazioni sui parametri ottenuti dalla cella servente. Durante ogni comunicazione, il modem cerca sempre di agganciarsi ad una cella nelle vicinanze, seppur la scelta della cella è una funzionalità implementata all'interno del protocollo del Modem stesso, è possibile riuscire ad ottenere le informazioni sulla cella a cui si è agganciati, tramite l'unione dei due comandi AT#MONI=0, che seleziona tra tutte le celle possibili che il modem riesce a vedere quella a cui è attualmente agganciato, e AT#MONI, che permette di visualizzare i dettagli relativi alla cella selezionata in precedenza.

Un esempio reale della map creata da questa API è il seguente:

```
map<string,string> CurrentCell = {

{"NetName" , "vodafone IT" },
{"Mode" , "4G"},
{"RSRP" , "-87"},
{"RSRQ" , "-12"},
{"TAC" , "3321"},
{"Id" , "000010A"},
{"EARFCN" , "1850"},
{"PWR" , "-57"},
{"DRX" , "64"}

}
```

Come è possibile vedere la map contiene varie informazioni come il tipo di segnale utilizzato (**4G**), il nome dell'operatore della cella a cui si è agganciati (**Vodafone IT**) e altri dati riguardanti la qualità e potenza del segnale come **RSRP** (Reference Signals Received Power), **RSRQ** (Reference Signal Received Quality) e **PWR/RSSI** (Received Signal Strength Indication).

6.5.4 getMapNeighbourCell

Input	***Output***	***AT command(s) used***
list<map<string,string» & boolean &	Void	AT#MONI=1, AT#MONI=2, AT#MONI=3, AT#MONI=4, AT#MONI=5, AT#MONI=6, AT#MONI=7, AT#MONI

Tabella 6.3: API getMapNeighbourCell

Questa API permette di costruire una struttura `std::list<map<string,string»` cioè una lista non ordinata di `std::map<string,string>` dove ogni map conterrà le informazioni di una singola cella seguendo il criterio di costruzione utilizzato per l'API precedente (6.5.3). Questa sarà la lista contenente tutte le informazioni sulle celle nelle vicinanze nel momento in cui si esegue la funzione.
Per ottenere questo genere di dato è stato necessario combinare tutti gli stati del comandi AT#MONI che possono essere configurati da 0 a 7 in base alle informazioni a cui si è interessati.
Iterativamente quindi si esegue il comando AT#MONI="x" seguito dal comdando AT#MONI per visualizzare lo stato. Le informazioni vengono salvate nella struttura `std::list<map<string,string»`.

6.5.5 getMapIndicatorControl

Input	***Output***	***AT command(s) used***
map<string, string> & boolean &	Void	AT+CIND?

Tabella 6.4: API getMapIndicatorControl

Questa API permette di costruire una struttura di tipo `std::map<string, string>` che conterrà al suo interno informazioni relativi a parametri di controllo del Modem, Un esempio può essere visto alla fine del Par. 6.3, in cui si mostra come è stata costruita e pensata la std::map risultante.
Questa API acquista un'importanza particolare grazie ad alcuni dei dati che dà, che sono risultati utilissimi al team di sviluppo. I parametri in questione sono: **"roam"** che permette di conoscere se la Sim si trova in uno stato di connessione roaming, **"call"** che segnala la presenza o meno di una connessione internet attualmente attiva e **"signal"** che permette di avere una stima sulla qualità del segnale.

6.5.6 getNetworkMode

Input	***Output***	***AT command(s) used***
string &	Void	AT+WS46?

Tabella 6.5: API getNetworkMode

Questa API è stata sviluppata per permettere di ottenere informazioni riguardo le configurazioni di *"switching"* di tecnologie di rete impostate nel Modem. Ogni connessione su rete cellulare infatti può essere stabilita per funzionare solo in una determinata modalità ed accettare solo tecnologie di rete ben definite e non accettarle tutte.
Esistono, per il **Modem Telit-Le920-EU**, 7 possibili modalità configurabili:

1. solo 2G
2. solo 3G
3. solo 4G
4. 2G e 3G
5. 2G e 4G
6. 3G e 4G
7. Tutte, 2G, 3G e 4G

6.5.7 setNetworkMode

Input	***Output***	***AT command(s) used***
int &	Void	AT+WS46="value"

Tabella 6.6: API setNetworkMode

Questa API permette di configurare la modalità e le regole che deve seguire il Modem circa lo *"switching"* tra diverse tecnologie di rete. Tramite questo modo può essere scelta una delle 7 possibili modalità elencate nel Par. 6.5.6. In questo modo durante le fasi di testing è stato possibile provare differenti scenari all'interno del progetto Long-Range V2X, permettendo al Team di valutare le prestazioni e il funzionamento del TIC-Framework in differenti ambienti di rete.

6.5.8 IsGood

Input	***Output***	***AT command(s) used***
boolean &	Void	None directly

Tabella 6.7: API IsGood

Questa API è stata richiesta dal Team di sviluppo per avere una valutazione sulla qualità del segnale ottenuto combinando differenti parametri scelti da loro e avere così una funziona che restituisse una determinato flag se qualità del segnale avesse rispettato delle soglie scelte a banco e modificabili in qualunque momento.
I parametri scelti per la valutazione della qualità variano in base al tipo di tecnologia con la quale si è connessi, strutturati come segue: **2G**: RSSI, RxQual, **3G**: RSSI, RSCP, EcIo, **4G**: RSSI, RSRP, RSRQ.
Per ogni tecnologia sono stati definiti **3 livelli** con cui valutare la qualità, elencati nelle tabelle seguenti:

2G	***Level 1***	***Level 2***	***Level 3***
RSSI	> -110	> -100	> -85
RxQual	< 7	< 6	< 5

Tabella 6.8: API IsGood: Soglie di valutazione per il 2G

3G	***Level 1***	***Level 2***	***Level 3***
RSSI	> -110	> -100	> -85
EcIo	> -15	> -12	> -9
RSCP	> -80	> -78	> -76

Tabella 6.9: API IsGood: Soglie di valutazione per il 3G

4G	***Level 1***	***Level 2***	***Level 3***
RSSI	> -95	> -85	> -75
RSRQ	> -20	> -18	> -15
RSRP	> -115	> -105	> -95

Tabella 6.10: API IsGood: Soglie di valutazione per il 4G

6.6 Utilizzo

Una volta sviluppata la **LibGSM** è stata inserita all'interno del Progetto del **Long-Range V2X** al fine di valutare le performance della rete stessa ed effettuare un'analisi contestuale dell'ambiente di Test nel quale il Team di sviluppo ha lavorato.
Per riuscire ad utilizzare le funzionalità richieste dal Team ho creato un applicativo allo stesso livello software degli **Use Cases**, elencati nel Cap. 4.3.3, chiamato per l'appunto **TestLibGSM**, che viene fatto partire insieme a tutto il resto del Framework effettuando le sue operazioni di analisi in perfetta armonia con le attività del **TIC-Framework**.

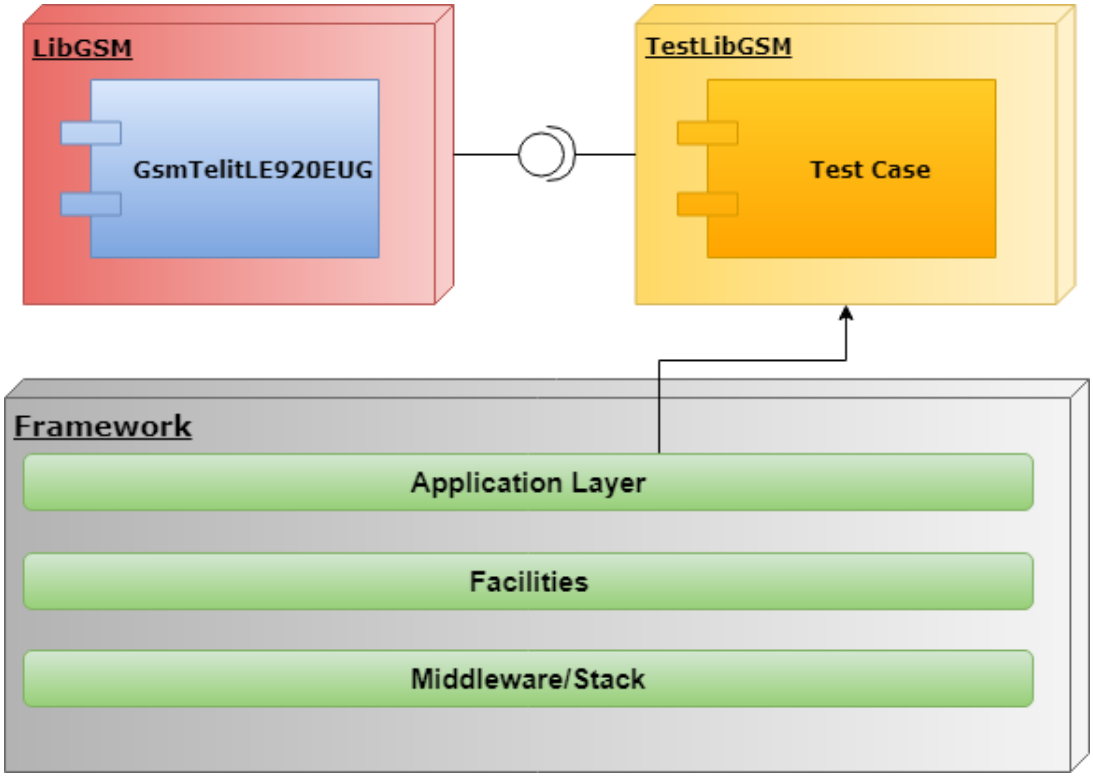

Figura 6.3: Architettura del TestLibGSM inserito nel Framework

In questo modo ho potuto effettuare i Test richiesti, di cui parlerò in maniera più approfondita nel Capitolo seguente, sfruttando l'ambiente sviluppato per le necessità del Framework. Ogni componente al suo interno è strutturata in modo da offrire servizi e funzionalità utili ad ogni altro componente.
Il **Test Case** sviluppato segue fedelmente il paradigma di programmazione con cui sono stati sviluppati gli altri Use Case con la sola eccezione di essere stato snellito da tutte quelle strutture specifiche che per questa fase risultavano inutili ed onerose.

Capitolo 7

Testing

7.1 Ambiente dei Test

Durante la fase di **Test** è stato scelto come ambiente quello in cui si è svolto una delle Demo sul **Long-Range V2X** dal team di sviluppo di **Technology Innovation - Connectivity and Cooperative Driving Projects** di **Magneti Marelli**.

Dopo aver completato quindi la fase di creazione dell'applicativo di Test, chiamato **TestCase**, Par. 6.6, ed aver configurato correttamente la Step3 per ottenere sia connettività sia robustezza nel mantenimento della connessione, Par. 5.6.2, ho integrato il mio TestCase all'interno del Framework ed iniziato ad effettuare tutta la parte relativa collaudo del mio lavoro.

Durante questa fase ovviamente sono stati condotti due tipi di Test:

- **Test a banco**: durante il quale l'applicativo è stato testato prima dal punto di vista della robustezza ed integrità, e poi dal punto di vista del funzionamento. Al fine di testare la bontà della mia Libreria e della corretta integrazione nel Framework. Durante questa fase sono stati decisi i vari parametri di analisi ed è stato identificato il pool di informazioni a cui si era maggiormente interessati durante la fase di Test su Veicolo.

- **Test su Veicolo**: durante il quale sono state tracciate, tramite dei file di `LOG`, tutte le informazioni che erano state decise insieme al Team, utili a loro per avere dei parametri di riferimento su cui poter effettuare delle analisi in previsione della Demo che da lì a poco sarebbe stata condotta.

Figura 7.1: In rosso sulla mappa: il percorso seguito durante la fase di Test e di Demo per il Long-Range V2X.

Il circuito di Test è stato ricavato proprio all'esterno della planimetria dello stabilimento di **Magneti Marelli** presente a **Venaria Reale (TO)**, Figura 7.1.

Durante la fase di preparazione inoltre è stato chiesto ad un operatore telefonico la possibilità di ottenere dei dati ottenuti dalle rilevazioni della propria compagnia. Si omette il nome dell'operatore per motivi di privacy, tuttavia la zona analizzata da questi riguarda proprio quella zona nella mappa della figura appena proposta. Comunque quei dati sono stati confrontati con quelli rilevati dal mio applicativo di Test, valutando pro e contro dei due approcci del tipo di dati ottenuti.

La zona interessa ai rilevamenti è quella presente nella Figura 7.1, ed è stata banco di prova anche per la prima **Demo** del **Long-Range V2X** tenutasi giorno **12 Dicembre 2018**, in cui per la prima volta sono state testate le funzionalità del Framework dopo la migrazione alla nuova tecnologia.

7.2 Obiettivi

Gli obiettivi di questi Test sono stati principalmente tre:

- Misurazione della qualità del segnale nell'area di interesse, ponendo attenzione nel confronto tra i dati raccolti e quelli offerti dall'operatore
- Confronto tra due operatori disponibili per scegliere il migliore sulla base di criteri prestabiliti
- Ottenere informazioni riguardanti la copertura della rete rispetto alla presenza delle celle nelle vicinanze

Sulla base di queste caratteristiche è stata scelta l'implementazione migliore del Test Case e le sue principali caratteristiche.

7.2.1 Test Case

Il Test Case è stato creato con le seguenti caratteristiche:

- Due Thread:
  - Il primo per ottenere le informazioni geografiche, latitudine, longitudine e heading, che vengono memorizzate ripetutamente in una struttura condivisa con il secondo thread.
  - Il secondo che effettua le chiamate alla **LibGSM** tramite le opportune API e dopo aver raccolto il pool di informazioni utili crea blocco nel file di **LOG** con tutti i dati registrati marcandoli con le coordinate geografiche presenti in quel momento nella struttura condivisa.
- Un ciclo continuo senza interruzioni nel quale ogni operazione viene ripetuta ad intervalli regolari. Presente sia nel primo thread per le informazioni Geografiche sia nel secondo che si occupa di reperimento dei dati.
- Memorizzazione in un file di **LOG** secondo un criterio ben stabilito nel quale ogni dato viene registrato seguendo il formato migliore in previsione di una seconda fase di **Post-Processing**.

7.3 Testing a banco

L'ambiente di lavoro con cui sono stati condotti i Test a Banco prevedeva l'utilizzo dei seguenti componenti:

- Step3
- Antenna GPS
- Antenna LTE
- PC/Laptop

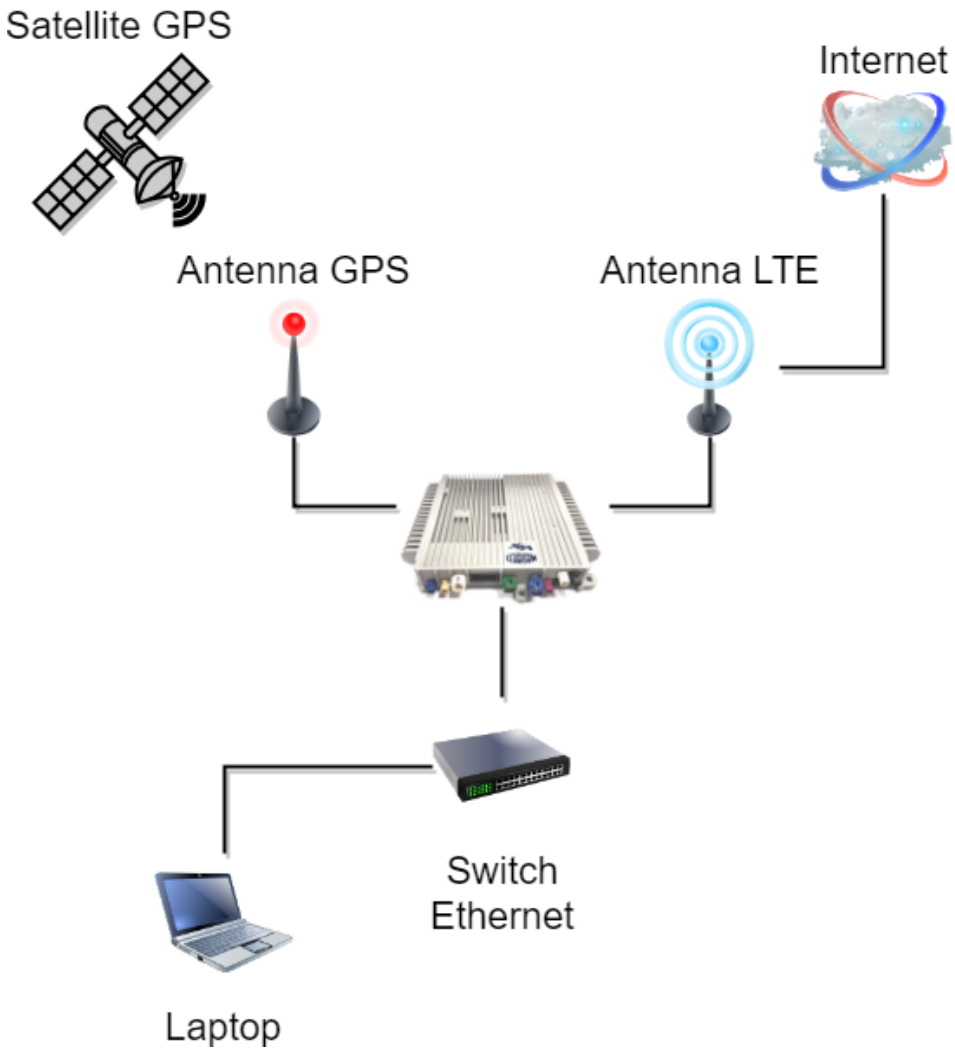

Figura 7.2: Ambiente di lavoro per i Test a banco

Durante la Fase di Testing a Banco sono stati effettuati i primi test sulla bontà dell'applicativo e l'integrazione sul Framework. Sono stati risolti alcuni bug e sono stati decisi i valori a cui si vuole essere interessati durante la fase di test.

I valori a cui ci si deve fare riferimento sono :

- NetName
- Mode
- RSSI/PWR
- DRX
- Cell-ID
- per il 2G:
  - BSIC
  - RxQual
  - LAC
  - ARFCN
  - TA
- per il 3G:
  - PSC
  - RSCP
  - UARFCN
  - EcIo
  - LAC
  - SCR
  - URA
- per il 4G:
  - RSRP
  - RSRQ
  - TAC
  - EARFCN

Una volta definite le API da utilizzare è stato deciso quindi il tempo di LOGGING a 2 secondi, nel quale vengono registrate le informazioni che sono state ritenute più utili ai fini del Test su Veicolo.

Durante questa fase è stato scelto il formato che deve tenere ogni unità informativa prima di essere registrata nel file di LOG per essere meglio analizzata durante il Post-Processing, di seguito ecco un esempio:

```
2018-11-08 13:55:50 - 007487: <trace> GPS:
latitude = 7.61860657,
longitude = 45.13959503,
heading = 96.06

2018-11-08 13:55:50 - 009534: <trace> Current Cell Information

2018-11-08 13:55:50 - 010222: <trace> GsmTest:
NetName =  I TIM
Mode = 3G
PSC = 510
RSCP = -83
LAC = EEB9
Id = 4595631
EcIo = -7.5
UARFCN = 10638
PWR = -76
DRX = 64
SCR = 8160
URA = 11134
```

Si è cercato di ottimizzare il tempo necessario al reperimento delle informazioni scegliendo quindi dei valori chiave e molto rappresentativi che ci hanno permesso di effettuare uno studio e un'analisi approfondita e sul circuito in previsione della Demo di Dicembre.

Conclusa questa fase condotta nei Laboratori dello stabilimento di Magneti Marelli di Venaria Reale (TO), una Step3 configurata da me con le modifiche apportate al Framework fatte per poter utilizzare il mio applicativo è stata porta su un veicolo aziendale con il quale è iniziata la seconda fase di Testing.

7.4 Testing su Veicolo

L'ambiente di lavoro con cui sono stati condotti i Test su Veicolo prevedeva l'utilizzo dei seguenti componenti:

- Step3
- Antenna GPS
- Antenna LTE
- CanBus
- Dashboard HMI
- Tablet HMI
- Switch Ethernet and Wifi Router
- PC/Laptop

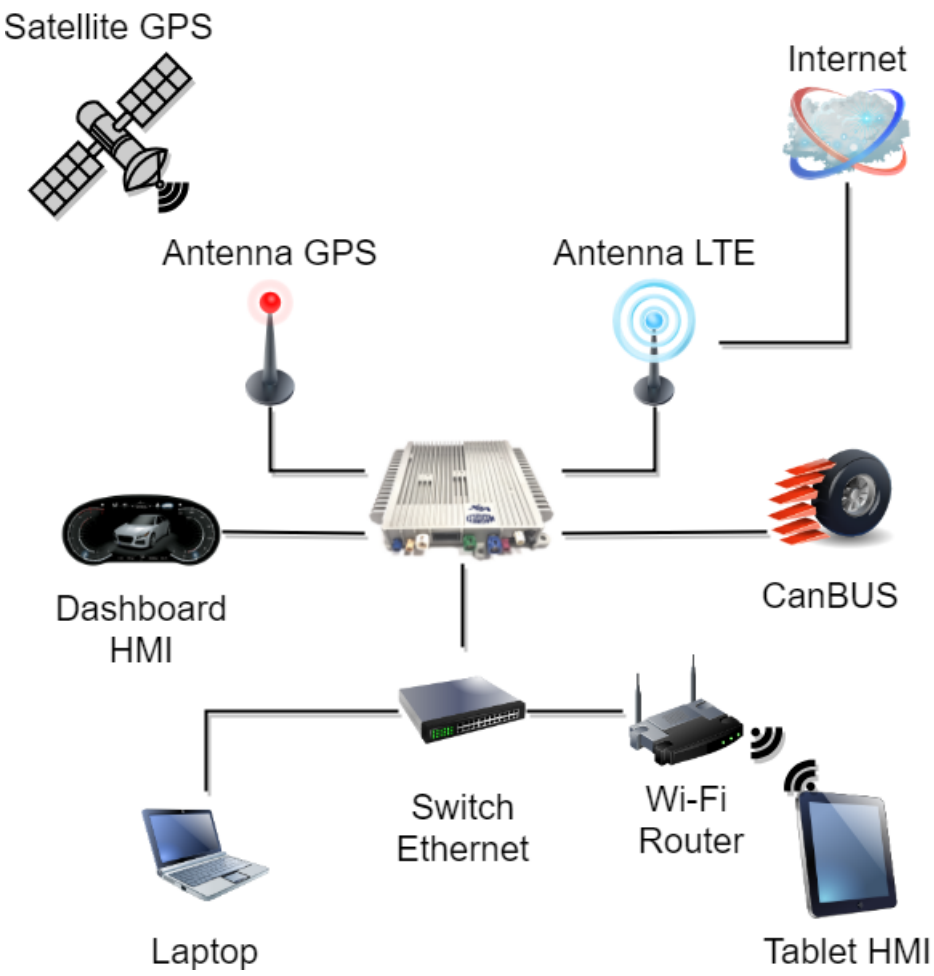

Figura 7.3: Ambiente di lavoro per i Test su veicolo

Durante la fase di test sono stati presi in considerazione i dati rilasciati da un operatore che per motivi di privacy chiameremo **"Operatore di riferimento"** e confrontati con i dati ottenuti da i due operatori che sono stati i fornitori ufficiali del progetto **Long-Range V2X** e che sono: **TIM** e **Vodafone**.

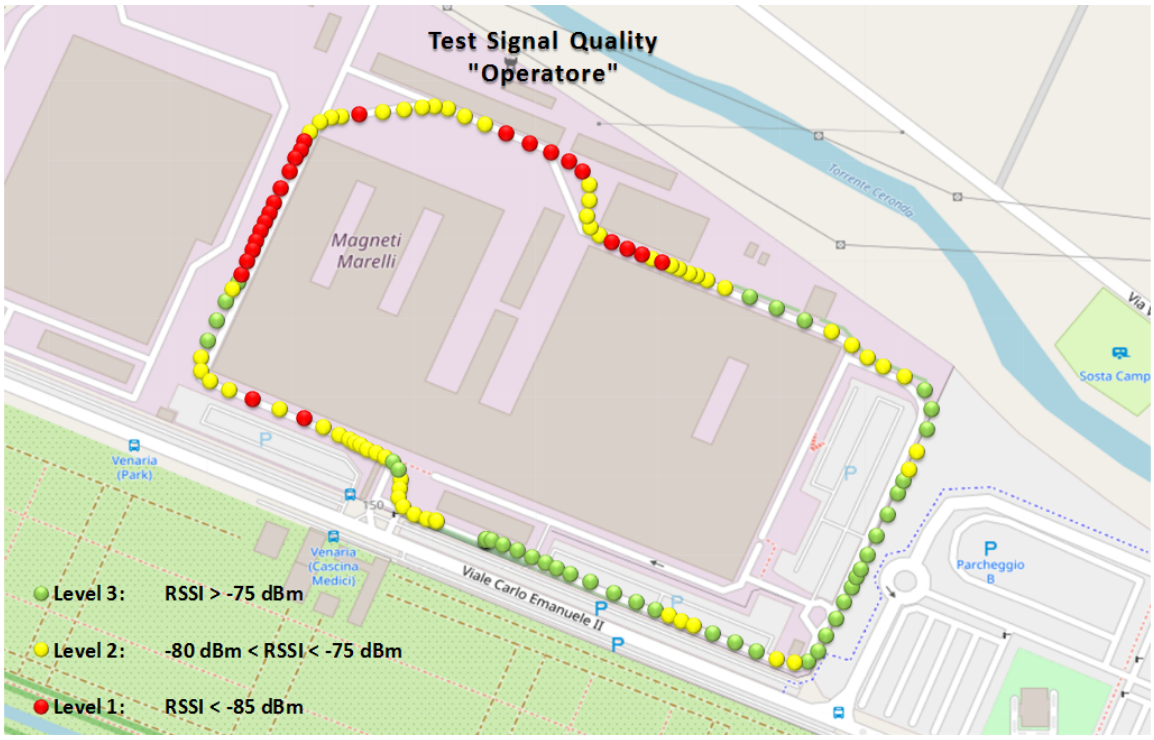

Figura 7.4: Copertura cellulare rilasciata dall"Operatore di riferimento" nella zona di nostro interesse

Come evidenziato in Figura 7.4, le zone in **Rosso** sono quelle a bassa copertura, dove cioè il segnale è più debole, quelle in **Giallo** sono a copertura media con una probabile qualità di poco oltre quella minima, quelle in **Blu** invece sono indicate come zone a buona copertura e in cui il segnale è stabile.

Durante la fase di Testing sono stati presi ad esame i due Operatori prima citati, Tim e Vodafone, e sono state effettuate analisi su:

1. **Tipo di Copertura**
2. **Alternanza Celle**
3. **Qualità del segnale**

7.4.1 Tipo di Copertura

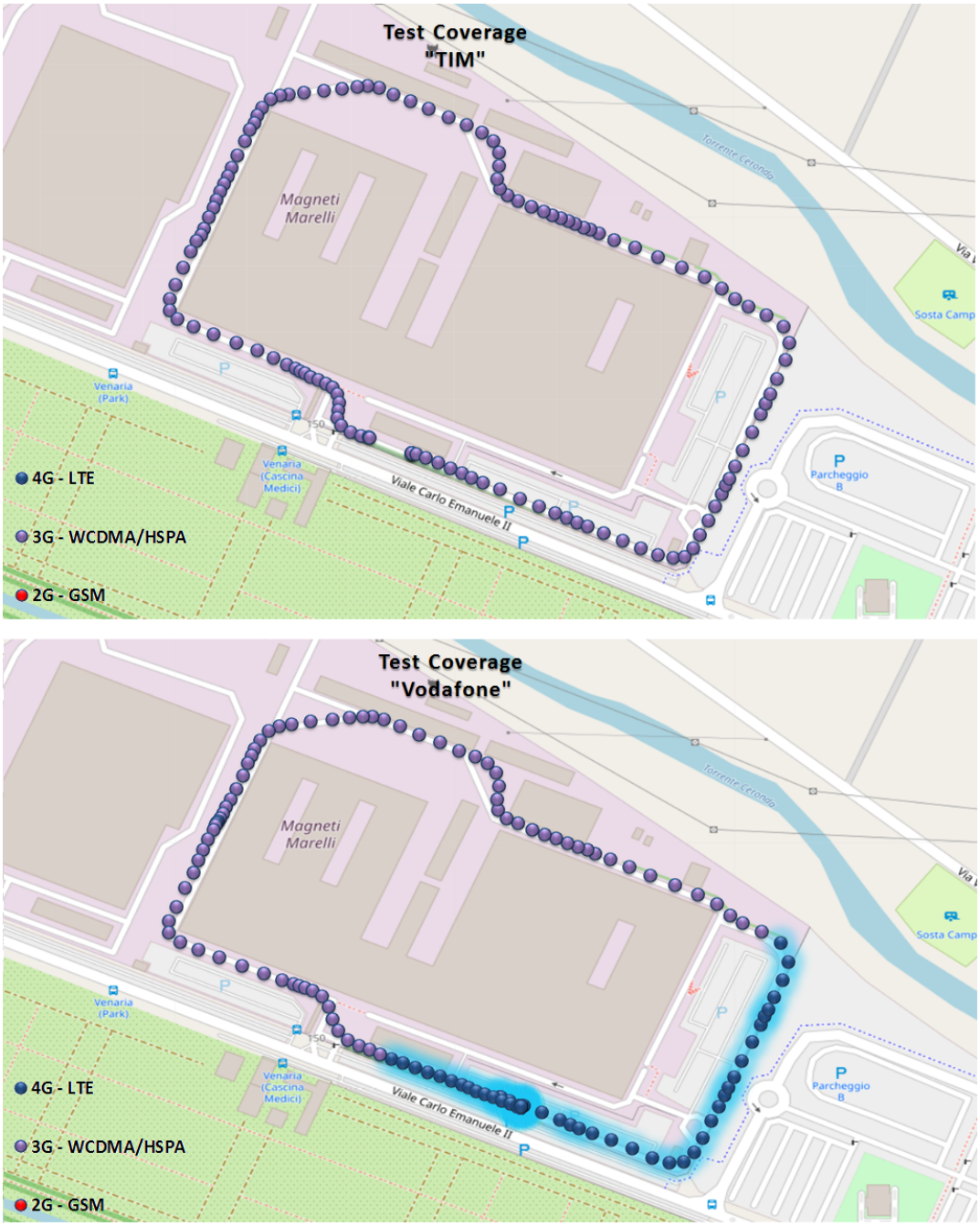

Figura 7.5: Test Copertura del Segnale cellulare nel Circuito con TIM e Vodafone

Il primo tipo di analisi effettuato è stato quello relativo alla tipologia di copertura presente nel Circuito, in particolare capire se e in quali punti la connessione LTE era utilizzabile e come poter sfruttare tutti i punti del tracciato nel migliore dei modi.

Nella Figura 7.5, è possibile vedere come la connessione 4G nella zona presa in analisi sia presente solo per Vodafone e solo per una porzione del Circuito.

La connettività TIM infatti risulta essere troppo debole per permettere una copertura 4G adeguata.
L'analisi è stata svolta configurando il modem in modalità di accettazione di qualunque tipo di rete, Vedere setNetworkMode() Par. 6.5.7 per chiarimenti, e segnare ogni cambiamento di tecnologia che in autonomia effettuava.

Nonostante però la connettività LTE fosse fornita solo da Vodafone, l'impossibilità di ottenere una copertura completa su tutto il Circuito ha dato modo di considerare anche altri parametri per la scelta dell'operatore.

Sono stati quindi svolti i Test per capire se questo dato avrebbe impattato ulteriormente sulle performance e soprattutto quanto riusciva ad essere efficiente un operatore rispetto ad un altro.
In particolare è stato interessante notare come nonostante una minore disponibilità cellulare la rete TIM è risultata essere la scelta migliore in previsione della DEMO fissata per Dicembre 2018.

Operator	***3G - WCDMA/HSPA***	***4G - LTE***
TIM	100%	0%
Vodafone	51%	49%

Tabella 7.1: Distribuzione Percentuale della copertura tra i due operatori sul Circuito

7.4.2 Alternanza Celle

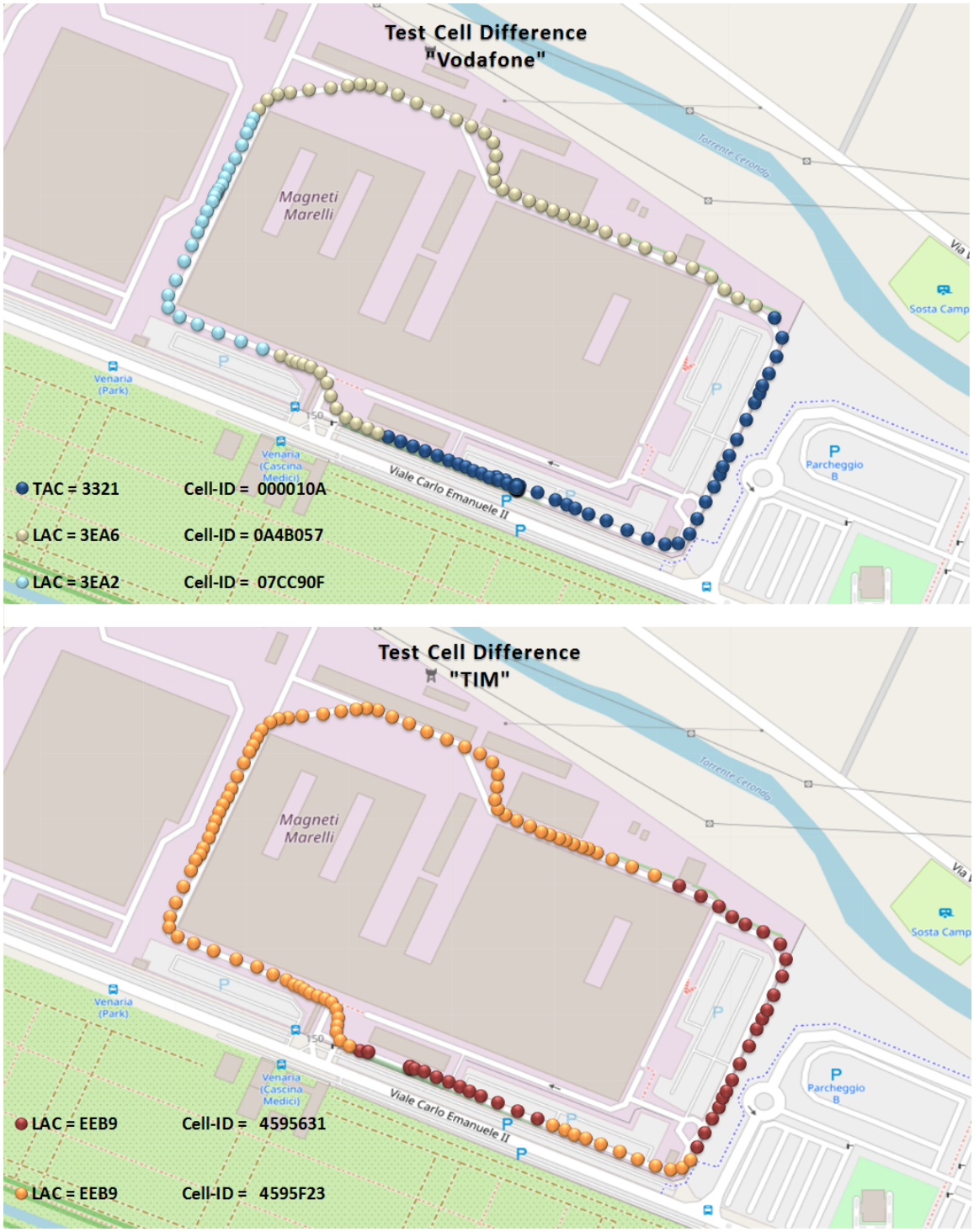

Figura 7.6: Test sui cambiamenti di Celle nel Circuito con TIM e Vodafone

Come evidenziato nella Figura 7.6, notiamo come in TIM l'alternanza di celle è limitato a sole due celle che tramite mapping sono state identificate rispettivamente a 600 metri e 1.65 Km, in Rosso in Fig. 7.7.

Analizzando Vodafone è evidente come l'alternanza di 3 celle, di cui una LTE in Arancio in Fig. 7.6, renda il segnale leggermente più instabile. Anche in questo caso è stato effettuato il mapping delle celle che si trovano rispettivamente a 715 metri, 1.56 Km e 2.6 Km, in Blu nella Fig. 7.7.

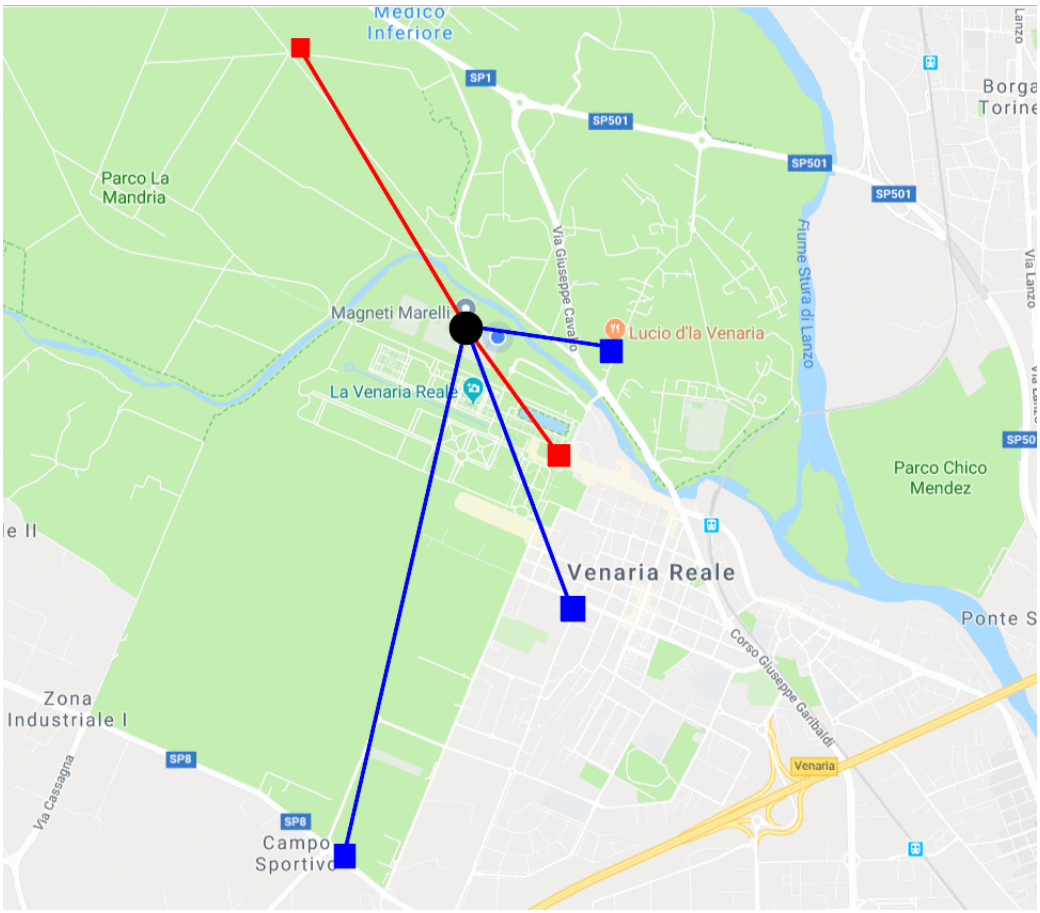

Figura 7.7: Posizione delle celle TIM (In Rosso) e Vodafone (In Blu) sulla zona di interesse [29]

7.4.3 Qualità del segnale

Figura 7.8: Test Qualità del Segnale cellulare nel Circuito con TIM e Vodafone

Nella Figura 7.8 la scala presentata suddivide la qualità del segnale in 3 Macro-aree, che sono simili ai tre livelli implementati nella funzione IsGood(), Cap. 6.5.8. Entrambi i due operatori presentano delle zone comuni di Qualità del segnale, il che potrebbe essere significativamente influenzato dalla conformazione del territorio e degli impedimenti fisici delle strutture vicine.

Tuttavia un'analisi sulle differenze la si percepisce sulla percentuale di rilevazioni delle due zone in base alla qualità del segnale.

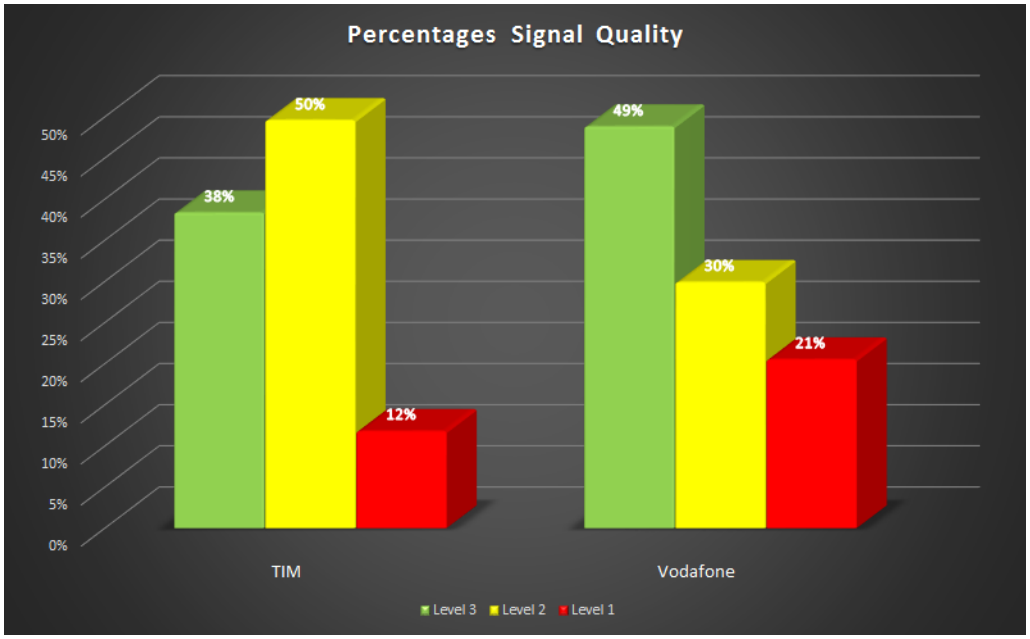

Figura 7.9: Percentuali sulla Qualità del Segnale nel Circuito con TIM e Vodafone

Come evidenziato in Figura 7.9, possiamo notare come la qualità del segnale di TIM nonostante sia molto altalenante riesce a coprire un buon 88%, sommando Level 3 + Level 2, del Circuito mantenendo una connettività non entusiasmante ma nella norma.

Il segnale Vodafone invece ha un'incredibile copertura di Level 3 (Qualità Massima) che raggiunge il 49% del tragitto a discapito però di un abbassamento della qualità del segnale nelle altre zone. Infatti evidenzia un 21% di scarsa copertura contro il 12% di TIM.

7.5 Valutazioni

Una volta effettuati questi Test è stata stabilita una **Formula di Valutazione** delle potenzialità di un Operatore nella zona da me analizzata al fine di stabilire quale dei due utilizzare per la Demo.
In tale senso l'Equazione è stata pensata per ottenere un valore negativo in presenza di reti in 2G o con bassa copertura, questa scelta è stata effettuata al fine di penalizzare molto di più la mancata, o scarsa, presenza di connettività, riuscendo comunque a valorizzare un tipo di connessione più stabile e performante e con una copertura cellulare molto più ampia.

La **Formula** è la seguente:

$$\mathcal{V} = \%4\mathcal{G} * \alpha \ + \ \%3\mathcal{G} * \beta \ + \ \%2\mathcal{G} * \gamma \ + \ \%\mathcal{L}3 * \mathfrak{a} + \%\mathcal{L}2 * \mathfrak{b} + \%\mathcal{L}1 * \mathfrak{c} \qquad (7.1)$$

Dove:

- $\%4\mathcal{G}, \%3\mathcal{G}, \%2\mathcal{G}$ sono le percentuali di presenza della copertura analizzati nel Par. 7.4.1.
- α, β, γ sono i pesi associati alle percentuali di presenza della copertura.
- $\%\mathcal{L}3, \%\mathcal{L}2, \%\mathcal{L}1$ sono le percentuali di qualità del segnale analizzati nel Par. 7.4.3.
- $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}$ sono i pesi associati alle percentuali di qualità del segnale.

I pesi scelti per le variabili sono stati i seguenti:

$$\alpha = 0,6 \qquad \beta = 0,4 \qquad \gamma = -1 \qquad (7.2)$$

$$\mathfrak{a} = 0,4 \qquad \mathfrak{b} = 0,4 \qquad \mathfrak{c} = -1 \qquad (7.3)$$

I dati ottenuti dai vari Test sono stati:

Operator	***4G***	***3G***	***2G***	***L3***	***L2***	***L1***
TIM	0%	100%	0%	38%	52%	12%
Vodafone	49%	51%	0%	49%	30%	21%

Tabella 7.2: Percentuali di Copertura sul Circuito di riferimento

Applicando quindi la Formula 7.1 con i dati descritti nella Tabella 7.2 e i pesi elencati nella 7.2 e 7.3 si ottengono i seguenti risultati:

$$\mathcal{V}_{TIM} = 0 * 0.6 + 1 * 0.4 - 0 * 1 + 0.38 * 0.4 + 0.5 * 0.4 - 0.12 * 1 = \boxed{\mathbf{0.632}}$$

$$\mathcal{V}_{Vodafone} = 0.49 * 0.6 + 0.51 * 0.4 - 0 * 1 + 0.49 * 0.4 + 0.3 * 0.4 - 0.21 * 1 = \boxed{\mathbf{0.604}}$$

Confrontando i due Valori ottenuti dalla Formula di Valutazione vediamo come la copertura TIM, ottenendo un valore di $\mathbf{V} = \mathbf{0.632}$, sembri essere leggermente migliore sul Circuito preso in esame rispetto a quella di Vodafone che ha ottenuto un valore di $\mathbf{V} = \mathbf{0.604}$.

Con questa formula di valutazione infatti possiamo ottenere una **V** che può avere un minimo di **-2** nel caso avessimo un Circuito con copertura **2G o nulla** su tutto il perimetro e con qualità del segnale classificata come **Level 1**, e un massimo di **1** nel caso invece avessimo una copertura totale in **4G** con una qualità del segnale tra il **Level 3 e Level 2**.

Questi dati possono essere giustificati dall'aver voluto effettuare il confronto valutando maggiormente l'ampiezza e completezza della copertura rispetto alle performance della rete.

Dopo quindi tutti i **Test** effettuati e dopo una fase di **Post-Processing** dei dati rilevati è stato possibile scegliere la copertura **TIM** come soluzione migliore per la **DEMO** prevista per Dicembre 2018.

Capitolo 8

Conclusioni e Sviluppi Futuri

8.1 Conclusioni

In conclusione, il mio lavoro di Tesi mi ha portato a confrontare differenti tecnologie, DSRC e C-V2X, potendone apprezzare sia i pregi che i difetti dell'una o dell'altra e riuscendo ad analizzare con attenzione tutti gli aspetti legati alla connettività cellulare.

I test descritti nel Cap. 7 hanno permesso di scegliere il miglior setup per effettuare la DEMO prevista per Dicembre 2018. Il Team è riuscito ad essere molto specifico con l'operatore nel segnalare le celle principali che sarebbero state utilizzate e chiedendone un potenziamento per le date indicate. Le mappe utilizzate sono state prese da una fonte Open Source quale **OSM, Open Street Map** [30].

I test hanno permesso di mettere in evidenza tutti gli aspetti positivi e negativi relativi alla scelta dell'operatore, della tecnologia e del luogo, permettendo di ottenere una visione più ampia.

Grazie a questi Test infatti la valutazione non è stata fatta semplicemente su valori come il Packet-Loss o il Throughput visto che in ambito di comunicazione veicolare hanno un'importanza differente nel momento in cui si vuole testare la bontà di una zona di interesse piuttosto che le performance, che sono più legate alla tecnologia utilizzata.

Tutto questo è stato possibile utilizzando solo una parte delle API da me sviluppate, in futuro altre API potrebbero servire per i più svariati ambiti, non solo legati all'analisi di un tracciato ma anche legati alle scelte che un veicolo deve compiere a fronte di informazioni legate alla connettività.

8.2 Evoluzioni Future della LibGSM

Pensando a quello che dovrebbe essere il futuro di questo progetto, possiamo dire che questa può essere considerata la base di una libreria molto più completa ed arricchita. Le potenzialità sono molteplici ed analizzando il contesto di sviluppo le possibilità sono tantissime.

Potremmo vedere infatti questa libreria come la base sulla quale costruire nuovi Software di sviluppo utili per differenti scopi, senza contare comunque la possibilità di ampliare enormemente il pool di API di base rispetto a quelle che fino ad ora sono state sviluppate.

Guardando il panorama del mondo **C-V2X** la **LibGSM** può essere la base per tre diversi macro-settori di sviluppo, quali:

- **Smart API**
- **Decision-Making Software**
- **Network Analyzer Software**

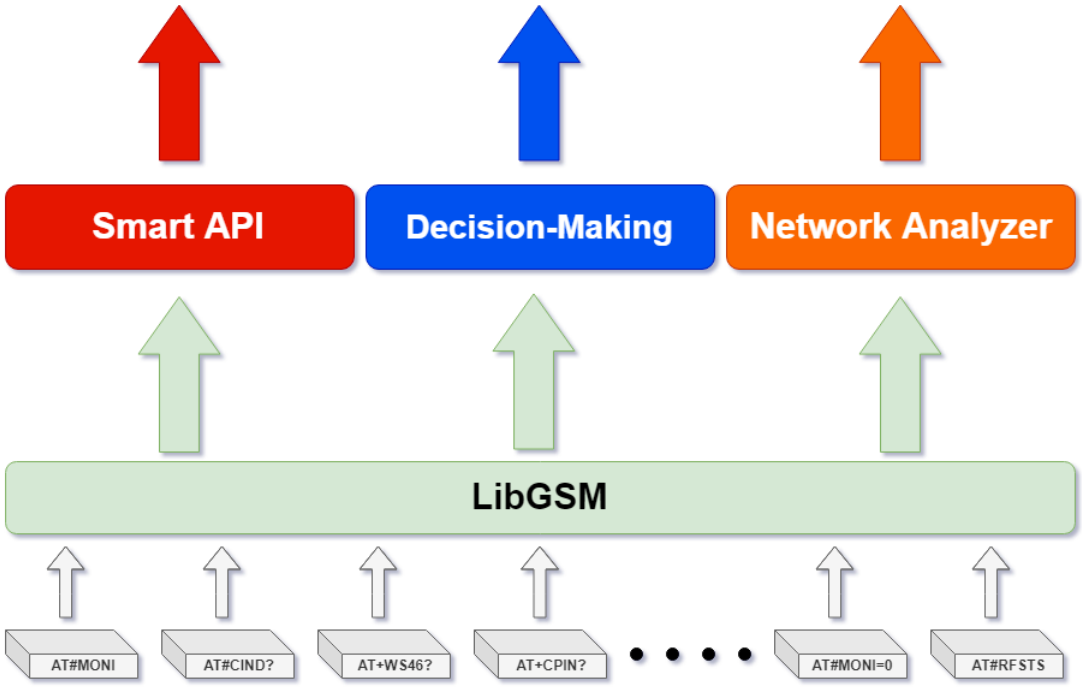

Figura 8.1: Esempi di software da sviluppare basati sulla LibGSM

8.2.1 Smart API

Uno dei possibili sviluppi futuri si basa sulla creazione di una Libreria di livello più alto in cui ogni funzione include al suo interno più API della **LibGSM** al fine di creare una nuova API più complessa per servizi specifici.

In questo livello si possono creare API che tramite l'immissione di un minimo livello di intelligenza artificiale creino nuovi modelli progettuali di analisi, oppure che offrano servizi software ai componenti di livello applicativo.

Le possibilità di sviluppo si ampliano se si pensa alle varie possibilità di:

- comporre API al fine di offrire funzioni di alto livello per eseguire compiti di configurazione in totale autonomia.

- Creare Servizi Software basati sull'instaurazione/configurazione/mantenimento della connessione di rete

- Ampliare il pool di funzionalità unendo più API per ottenere informazioni più precise, specifiche e inerenti

8.2.2 Decision Making Software

Un **Decion-Making Software (Software DM)** è un tipo di programma per applicazioni informatiche che aiuta individui e organizzazioni a fare scelte e prendere decisioni, in genere classificando, dando la priorità o scegliendo tra una serie di opzioni.

La maggior parte del software DM si concentra sulla classificazione, la priorità o la scelta tra alternative caratterizzate da più criteri o attributi. Pertanto, la maggior parte dei software DM si basa sull'analisi decisionale, di solito multi-criterio, e viene spesso definita **Decision Analysis** o **Multi-Criteria Decision-Making**, comunemente abbreviato in **Decion-Making Software**. Alcuni sistemi di supporto alle decisioni includono un componente software DM.

Nel contesto del **C-V2X** un Software DM potrebbe basare le proprie scelte su vari parametri come la qualità della connessione, la distanza dai una cella o vari stati di configurazione del Modem. Ogni decisione può essere fatta per decidere cosa inviare, quando e in che quantità.
Inoltre si avrebbe la possibilità di classificare dati, parametri e informazioni utili run-time al fine di ottimizzare il lavoro di ogni OBU (On-Board Unit).

8.2.3 Network Analyzer Software

Un **Network Analyzer Software** è un programma pensato per effettuare delle rilevazioni di dati, analizzarle e restituire in output un risultato ben preciso, dipendente dallo scopo per cui è stato pensato.

Un **Network Analyzer Software** è un definizione molto generica, infatti esistono vari software di questo tipo che si specializzano per un livello ben specifico con funzionalità uniche e distintive della tipologia di rete analizzata.
Un esempio in proposito sono i **Packet Analyzer Software** che possono intercettare e registrare il traffico che passa su una rete digitale o parte di una rete.

L'Analisi che si propone basata sulla **LibGSM** è quella di una valutazione delle performance della rete basata su parametri relativi alla connettività che possono essere: **RSSI, RSRP, RSRQ**. Senza dimenticare anche altri parametri quali le coordinate geografiche e la posizione fisica della cella rispetto alla propria posizione. In questo modo si potrebbe automatizzare anche il meccanismo del Post-Processing al fine di ottenere dei dati di rilevanza statistica per il tipo di rete che si vuole analizzare.

8.3 Sviluppi Futuri per il C-V2X

Pensando allo sviluppo lato C-V2X le possibilità sono molteplici. La tecnologia infatti è ancora giovane e si appresta a far partire un nuovo modo di intendere il mondo delle autovetture.
In tal senso lo sviluppo principale si concentrerà sul mondo 5G (Fifth Generation) e su tutte le innovazioni che porterà sia sul lato delle telecomunicazioni che da quello dei servizi offerti.

Le auto di domani non saranno soltanto più intelligenti, ma anche estremamente connesse tra loro e con l'ambiente che le circonda, tutto questo avrà un impatto enorme sulla sicurezza stradale e più in generale sulla qualità e la quantità del tempo che trascorriamo nel traffico.

Con l'interfaccia PC5 le comunicazioni tramite 5G cambieranno il nostro modo di intendere la connettività, si avrà infatti possibilità di comunicare in modo diretto con tutti gli elementi della rete, il suo punto forte sarà anche quello di poter scambiare dati anche in assenza di copertura cellulare, ad esempio nelle zone montuose o poco servite dagli operatori telefonici.

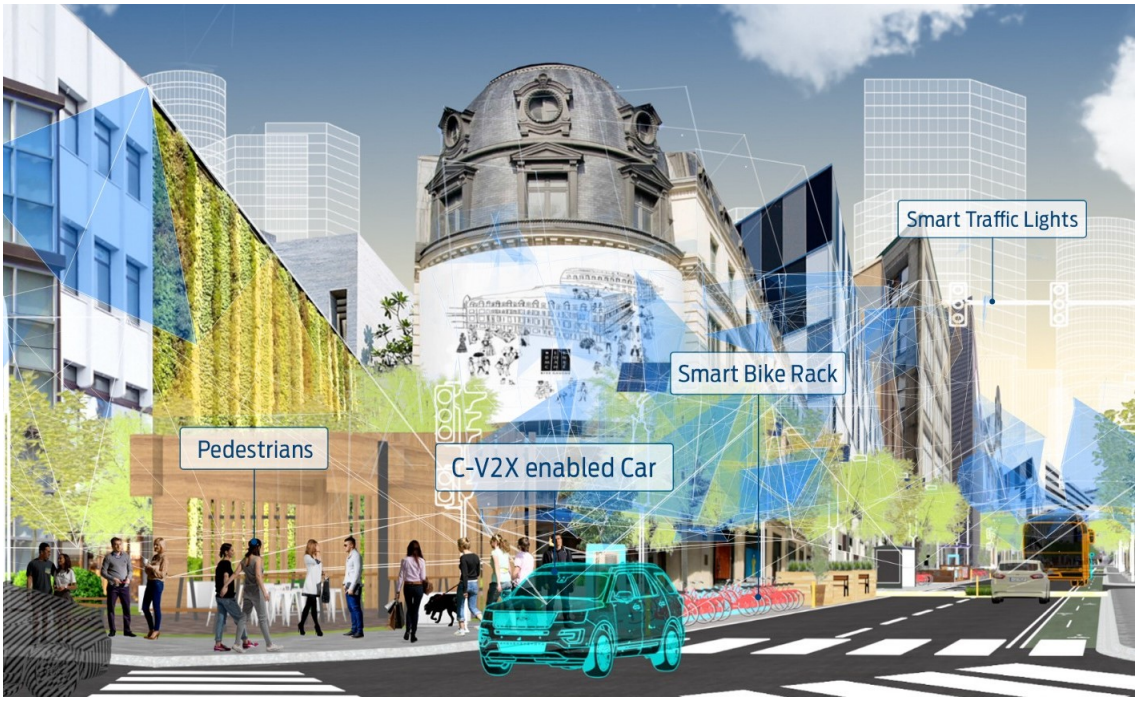

Figura 8.2: C-V2X, una tecnologia che guarda al futuro

Il lavoro che ho svolto quindi nell'ambito della comunicazione cellulare potrà essere continuato ed ampliato aggiungendo tutte le nuove funzionalità che la nuova tecnologia ci permetterà di utilizzare.
Da un incredibile aumento delle performance della rete ad una maggiore completezza dal punto di vista delle connessioni, il 5G e il C-V2X sembrano portarci verso un futuro più dinamico, più connesso e soprattutto più sicuro. Estendendo non solo incredibili possibilità per noi sviluppatori ma anche per ogni persona che usufruirà dei servizi che di volta in volta riusciremo ad offrire.

Bibliografia

[1] Chai K Toh. " Ad Hoc Mobile Wireless Networks: Protocols and Systems, Prentice Hall". In: *Pearson Education* (dic. 2001).

[2] *Motor Vehicle Registered in the world*. URL: https://www.ceicdata.com/en/indicator/iran/motor-vehicle-registered (visitato il 16/04/2019).

[3] *Dati ISTAT sugli Incidenti Stradali in Italia nel 2016*. URL: https://www.istat.it/it/archivio/202802 (visitato il 16/04/2019).

[4] *Magneti Marelli S.p.a.* URL: https://www.magnetimarelli.com/it/azienda (visitato il 16/04/2019).

[5] M. L. Sichitiu e M. Kihl. "Inter-vehicle communication systems: a survey". In: *IEEE Communications Surveys Tutorials* 10.2 (feb. 2008), pp. 88–105. ISSN: 1553-877X. DOI: 10.1109/COMST.2008.4564481.

[6] 3GPP. *Vehicle-to-Everything (V2X) services based on LTE; User Equipment (UE) radio transmission and reception*. Technical Report (TR) 36.786. Version 14.0.0. 3rd Generation Partnership Project (3GPP), mar. 2017. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3104.

[7] D. Jiang e L. Delgrossi. "IEEE 802.11p: Towards an International Standard for Wireless Access in Vehicular Environments". In: *VTC Spring 2008 - IEEE Vehicular Technology Conference*. Mag. 2008, pp. 2036–2040. DOI: 10.1109/VETECS.2008.458.

[8] S. Gräfling, P. Mähönen e J. Riihijärvi. "Performance evaluation of IEEE 1609 WAVE and IEEE 802.11p for vehicular communications". In: *2010 Second International Conference on Ubiquitous and Future Networks (ICUFN)*. Giu. 2010, pp. 344–348. DOI: 10.1109/ICUFN.2010.5547184.

[9] "IEEE Standard for Wireless Access in Vehicular Environments (WAVE) – Networking Services". In: *IEEE Std 1609.3-2016 (Revision of IEEE Std 1609.3-2010)* (apr. 2016), pp. 1–160. DOI: 10.1109/IEEESTD.2016.7458115.

[10] Q. Chen, D. Jiang e L. Delgrossi. "IEEE 1609.4 DSRC multi-channel operations and its implications on vehicle safety communications". In: *2009 IEEE Vehicular Networking Conference (VNC)*. Ott. 2009, pp. 1–8. DOI: 10.1109/VNC.2009.5416394.

[11] *Qualcomm Presentation C-V2X*. URL: https://www.qualcomm.com/invention/5g/cellular-v2x (visitato il 16/04/2019).

[12] *The Case for Cellular V2X for Safety and Cooperative Driving*. URL: http://5gaa.org/wp-content/uploads/2017/10/5GAA-whitepaper-23-Nov-2016.pdf (visitato il 16/04/2019).

[13] *Qualcomm C-V2X documentation*. URL: https://www.qualcomm.com/media/documents/files/accelerating-c-v2x-commercialization.pdf (visitato il 16/04/2019).

[14] *Use Case - Pedone, punto cieco*. URL: https://www.pcprofessionale.it/tech/automotive-tecnologia-cv2x/ (visitato il 16/04/2019).

[15] *Evolution, from DSRC and ITS-G5 to LTE-V2X*. URL: http://www.nvp.co.kr/news/articleView.html?idxno=121979 (visitato il 16/04/2019).

[16] *xE920 Family Automotive*. URL: https://www.telit.com/m2m-iot-products/cellular-modules/automotive-modules/xe920-family/ (visitato il 16/04/2019).

[17] *The KDevelop Project*. URL: https://web.archive.org/web/20030621132836/http://www.kdevelop.org/index.html?filename=main1999.html (visitato il 16/04/2019).

[18] Dr. Gerry Meyer. *The PPP Encryption Control Protocol (ECP)*. RFC 1968. Giu. 1996. DOI: 10.17487/RFC1968. URL: https://rfc-editor.org/rfc/rfc1968.txt.

[19] *Overview on PPP and its benefits*. URL: http://www.tcpipguide.com/free/t_PPPOverviewHistoryandBenefits-2.htm (visitato il 16/04/2019).

[20] William A. Simpson. *The Point-to-Point Protocol (PPP)*. RFC 1661. Lug. 1994. DOI: 10.17487/RFC1661. URL: https://rfc-editor.org/rfc/rfc1661.txt.

[21] *Linux man page - PPPD*. URL: https://linux.die.net/man/8/pppd (visitato il 16/04/2019).

[22] *PPP Cisco Documentation*. URL: https://www.cisco.com/c/en/us/support/docs/wan/point-to-point-protocol-ppp/25440-debug-ppp-negotiation.html (visitato il 16/04/2019).

[23] *The Internet Engineering Task Force (IETF)*. URL: https://www.ietf.org/ (visitato il 16/04/2019).

[24] *Dynamic DNS*. URL: https://dyndns.it/ (visitato il 16/04/2019).

[25] *The Modem of Dennis Hayes and Dale Heatherington*. URL: http://history-computer.com/ModernComputer/Basis/modem.html (visitato il 16/04/2019).

[26] *More Modem Commands*. URL: http://www.chebucto.ns.ca/Help/Dialup/MoreModemCommands.shtml (visitato il 16/04/2019).

[27] 3GPP. *AT command set for User Equipment (UE)*. Technical specification (TS) 27.007. Version 15.4.0. 3rd Generation Partnership Project (3GPP), dic. 2018. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=1515.

[28] *Telit LE920A4/LE910C1 - AT Commands Reference Guide*. URL: https://www.telit.com/wp-content/uploads/2017/09/80490ST10778A_AT_Commands_Reference_Guide_LE9x0A4_LE910Cx__R2.pdf (visitato il 16/04/2019).

[29] *Google Maps (n.d.). Magneti Marelli, Venaria Reale (TO) retrived from*. URL: https://goo.gl/maps/qwQqS2U45t12 (visitato il 16/04/2019).

[30] *Open Street map (OSM). Magneti Marelli, Venaria Reale (TO)*. URL: https://www.openstreetmap.org/#map=17/45.14089/7.61906 (visitato il 16/04/2019).