ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

CAMPUS DI CESENA
SCUOLA DI INGEGNERIA E ARCHITETTURA

Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

COORDINAZIONE DI VEICOLI AUTONOMI: SIMULAZIONE DI INCROCI STRADALI

Elaborata nel corso di: Sistemi autonomi

Tesi di Laurea di:
MATTIA BORRILLO

Relatore:
Prof. OMICINI ANDREA

Co-relatori:
Prof. Mariani Stefano

ANNO ACCADEMICO 2017-2018
SESSIONE II

PAROLE CHIAVE

Autonomia

Internet of Vehicles

Coordinazione

V2V

V2I

Alla mia famiglia.

Indice

Introduzione

In un contesto in cui l'evoluzione tecnologica ha portato alla luce un insieme di possibilità all'avanguardia per la realizzazione di reti wireless sfruttando l'esponenziale diffusione di terminali senza fili, è cosa del tutto naturale che questo vento di cambiamento e progresso arrivasse a influenzare anche lo sviluppo del trasporto privato urbano. L'escalation nella diffusione di dispositivi computazionali evoluti ha determinato una vera e propria metamorfosi sociale: lo stile di vita delle persone ha subito bruschi cambiamenti, i quali, studiati con il senno di poi, sembravano quasi impossibili a priori. ora la sfida è riposta su una nuova frontiera, quella dei trasporti e del traffico cittadino. Quest'ultimo è in sempre più rapida ascesa come problema che causa in primis un decremento produttivo importante e un aumento del livello di stress medio per individuo, scatenando una serie di conseguenzeche peggiorano la qualità di vita dei residenti nei centri urbani, oltre che danni all'ambiente con un consumo medio di carburante in costante aumento.
Di pari passo allo sviluppo tecnologico legato alla comunicazione è avvenuto il miglioramento delle apparecchiature che usano tali tecnologie wireless, riuscendo a portare il tutto ad un livello che permette il raggiungimento della cosiddetta *situatedness* dei dispositivi. Il concetto di situatedness si riferisce alla consapevolezza di un sistema computazionale nei riguardi dell'ambiente circostante e si riflette nella percezione dei cambiamenti che avvengono e nella capacità di determinarne. Questo documento si interesserà del caso di studio riguardante le automobili autonome e i modelli di coordinazione che aspirano a migliorare sia la sicurezza che le performance in termini di lunghezza media di viaggio in presenza di fenomeni di traffico congestionato cittadino.
L'automobile nasce come mezzo di trasporto privato di lusso circa un secolo fa, ma da allora l'evoluzione delle tecniche produttive e la necessità di effettuare spostamenti rapidi per riuscire ad adempiere agli impegni personali

di ogni individuo hanno reso questo bene un bene di prima necessità accessibile alla quasi totalità della popolazione. Questo fenomeno di eccessiva penetrazione del mercato dell'automobile all'interno del tessuto sociale ha di fatto prodotto un risultato contrario alle aspettative: il numero di mezzi in circolazione sulle strade urbane supera di gran lunga quella che è la loro capacità. Il traffico è un fenomeno parte integrante della vita urbana, un fenomeno che diventa in molti casi una piaga sociale sottraendo tempo alle relazioni sociali e al quieto vivere dei soggetti coinvolti. Essi accusano problemi di accumulo di stress subendo danni alla salute in maniera diretta, danni che si differenziano nel tipo ma non nella gravità rispetto ai cosiddetti "costi esterni" cioè gli effetti negativi pagati dalla comunità (anche da chi un auto nemmeno la possiede) a causa del sistema di trasporto automobilistico in termini di inquinamento acustico ed atmosferico. Oltre a ciò esistono i pericoli dovuti alla sicurezza stradale che molto spesso non riesce ad essere garantita provocando decessi tra gli automobilisti e non. In Italia il picco si è toccato nell'anno 2016 quando si sono verificati 175.191 incidenti stradali con lesioni a persone che hanno provocato 3.283 vittime e quasi 250.000 feriti (fig. 1). Le indagini dell'istituito italiano di ricerca concludono che le cause maggiori che provocano circostanze di incidente sono da attribuire ai seguenti fattori:

- Guida distratta o indecisa
- Mancato rispetto dei segnali di precedenza e dei semafori
- Velocità al di sopra dei limiti di sicurezza

A causa dei problemi emersi nel tempo, il trend di sviluppo automobilistico ha fissato dei target differenti rispetto al passato: infatti ora non sono più la ricerca delle prestazioni e dell'estetica raffinata gli obiettivi che permettono di arrivare ad avere una posizione dominate all'interno del mercato alle case produttrici bensì l'avanguardia tecnologica, l'integrazione con i dispositivi di uso comune come gli smartphone e i sistemi autonomi di supporto alla guida ($ADAS$) concentrano gli sforzi di investimento in maniera molto decisa. Sono questi i presupposti in cui nasce l'idea di un futuro caratterizzato da un insieme di reti di veicoli connessi caratterizzati da una guida autonoma le cui interazioni (non banali) daranno vita ad un sistema complesso urbano. L'obiettivo è comprendere come la la creazione di reti

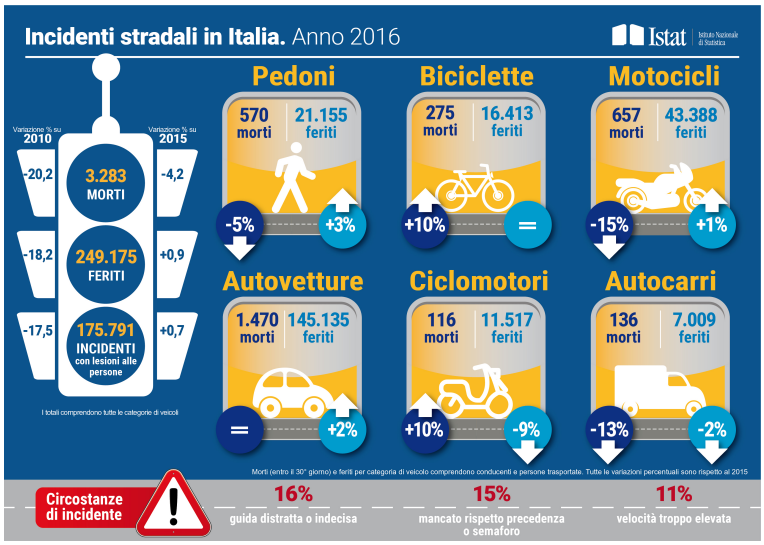

Figura 1: Circostanze incidenti stradali Italia 2016.

veicolari urbane, l'inclusione di elementi autonomi ed intelligenti all'interno dei veicoli e l'introduzione di protocolli standard per la coordinazione dei veicoli possa migliorare la situazione, si sta promuovendo un'integrazione tra diversi rami dell'ingegneria al fine di creare nuovi sistemi denominati *Intelligent Transport o Transportation Systems (ITS)*.
Per ITS s'intende:

> "l'integrazione delle conoscenze nel campo delle telecomunicazioni, elettronica, informatica - in breve, la "telematica" - con l'ingegneria dei trasporti, per la pianificazione, progettazione, esercizio, manutenzione e gestione dei sistemi di trasporto. Questa integrazione è finalizzata al miglioramento della sicurezza della guida e all'incolumità delle persone (safety), alla sicurezza e protezione dei veicoli e delle merci (security), alla qualità, nonché all'efficienza dei sistemi di trasporto per i passeggeri e le merci, ottimizzando l'uso delle risorse naturali e rispettando l'ambiente."

Capitolo 1

Tecnologie abilitanti

In questo capitolo saranno analizzati gli elementi che hanno portato alla possbilità di cominciare in ambito accademico ad affrontare i discorsi che riguardano la coordinazione dei veicoli autonomi. Sarà l'unico punto dell'elaborato in cui saranno elencate le tecnologie che costituiscono la base tecnologica che permette la realizzazione di automobili autonome. Successivamente sarà analizzata la coordinazione in ambito di traffico costituito da veicoli autonomi insieme alle caratteristiche dei protocolli. Assieme a questo verrà introdotto un ambiente per la simulazione di tali protocolli, i risultati ottenuti saranno poi messi in confronto con lo stato attuale della coordinazione veicolare al fine di trarre una prima conclusione sull'efficacia dei metodi introdotti.

1.1 Implementazione veicoli autonomi

I veicoli autonomi (chiamati anche *self-driving*, *driverless* o *robotic*) sono un obiettivo ed un sogno per i costruttori sin dai primi anni '50 quando case produttrici come Toyota ed Audi introdussero sistemi di supporto alla guida che includono il servo-sterzo fino ad arrivare al periodo di fine anni '80, primi anni '90 che segnano uno spartiacque molto importante dal punto di vista tecnologico [12]: le auto commerciali, inizialmente dedicate ad una fetta di mercato rivolta a soggetti in grado di spendere grandi somme di denaro, iniziano ad essere equipaggiate con dispositivi a comando totalmente autonomo. A tal riguardo bisogna per prima cosa definire cosa si intende per veicoli autonomi e quali sono le categorie in cui si possono suddividere

per meglio inquadrare il caso di studio e lo scenario in cui il tema della coordinazione tra veicoli sarà fondamentale per raggiungere gli obiettivi di ottimizzazione dei flussi di traffico. Le categorie prendono il nome di livelli e possono così essere descritte [10]:

- Livello 0: la dinamica di guida, il controllo sui comandi del veicolo e l'osservazione dell'ambiente circostante sono sotto il completo controllo dell'individuo.
- Livello 1: esiste una automatizzazione dei comandi che riguardano sistemi di cruise control o sistemi ABS che danno un supporto al guidatore rendendo più sicura la guida in caso di necessità.
- Livello 2: l'automatizzazione di livello 1 viene ulteriormente potenziata grazie alla presenza di tecniche che combinano il funzionamento di più sistemi in contemporanea.
- Livello 3: esistono sistemi di riconoscimento automatico del pericolo che comportano avvisi al conducente, il quale ha la responsabilità di riportare il veicolo in condizioni di sicurezza; è il primo livello di classificazione che comporta una capacità di riconoscimento automatico ed autonomo del contesto ambientale.
- Livello 4: la complessità dell'automatizzazione a questo livello aumenta e comporta l'abilità dei sistemi di supporto alla guida di riuscire a mantenere, proattivamente, in uno stato di sicurezza il veicolo agendo sui comandi del veicolo stesso per contesti ambientali di complessità pre-definita.
- Livello 5: l'autonomia totale nella conduzione del veicolo è catturata da questa classe; i sistemi che rispettano questi requisiti infatti riescono a percepire l'ambiente circostante in ogni circostanza e autonomamente reagiscono azionando i comandi di guida.

> "Si potrà raggiungere il livello 5 solo quando la vettura sarà integrata, mediante connessione con la rete, con l'intelligenza globale del sistema traffico" [19]

Ci troviamo in un periodo storico in cui l'escalation alla ricerca dell'innovazione ha portato all'investimento di grosse somme di denaro che dovranno

dare come risultato dei progetti pilota in grado di poter soddisfare i requisiti per i livelli 4 o 5.
Questo compito non è per nulla banale: il contesto nella quale un auto si trova ad operare è sensibile ad un numero di eventi totalmente imprevedibili e solo parzialmente descrivibile attraverso modelli di tipo statistico che possano essere affidabili. La sfida maggiore consiste nel riuscire a garantire degli standard di sicurezza tra la moltitudine di possibili interazioni che si devono manifestare attraverso la comunicazione tra agenti autonomi connessi in rete e non. La complessità maggiore è avere a disposizione moduli software che riconoscano in tempo reale oggetti animati e non nei paraggi, e per i primi avere un modello di classificazione predittivo che faccia adottare soluzioni preventive per affrontare e prevedere situazioni complesse da affrontare. Basta immaginare, per fare un paragone con la guida con conducente, la cautela con la quale un conducente di un veicolo si appresta ad affrontare un segmento di percorso dove nota la presenza di bambini o animali per comprendere la complessità del problema. Far operare un software all'interno di una strada pubblica, con l'obiettivo di condurre in sicurezza merci o persone da una partenza ad una destinazione è un compito arduo che se, misurato ad esempio, nella complessità del codice sorgente dei moduli software è per lo meno di un ordine di grandezza superiore perfino di quello di un Boeing 878 Dreamliner, senza considerare in questo la quasi impossibilità di testare la sua infallibilità.

Il sistema che permette la realizzazione di mezzi a guida autonoma è decisamente complesso e consiste nella maggior parte dei casi di almeno tre moduli che cooperano per riuscire ad arrivare a soddisfare i requisiti di livello 5:

- modulo algoritmico: si occupa di estrarre informazioni significative dai dati grezzi dei sensori per comprendere l'ambiente circostante e reagire prendendo decisioni immediate.

- sistema operativo: è il sotto-sistema che si occupa di garantire l'afflusso dei dati dai sensori al modulo algoritmico ed allo stesso tempo di eseguire le azioni pragmatiche che vengono indicate dagli output algoritmici. Un ulteriore compito di questo modulo è di far da garante sui requisiti che tutti i sistemi di tipo real-time necessitano, compresa l'affidabilità.

- piattaforma cloud: non è un modulo né fisico né logico effettivamente facente parte del veicolo stesso ma rappresenta una componente indispensabile per il suo corretto funzionamento. Non sarebbe infatti possibile avere un mezzo autonomo non connesso, e le piattaforme cloud forniscono ulteriore capacità di calcolo, modelli di training per i moduli decisionali e dati aggiornati per il tracking e la viabilità in tempo reale.

1.1.1 Soluzioni software per veicoli a guida autonoma

Il modulo computazionale rappresenta il core del veicolo, è responsabile come già detto delle attività di *sensing* e *decision making* lungo la tratta garantendo il raggiungimento della destinazione attraverso un percorso sicuro. Molti e di diverse tipologie sono i sensori: ogni tipologia porta con sé vantaggi e svantaggi tali per cui la precisione in questo tipo di compito può essere raggiunta attraverso lo sfruttamento di approcci combinati [13].

- Rilevamento posizione ed inerzia: i due sistemi maggiormente utilizzati sono il GPS *(Global Positioning System)* e l'IMU *(Inertial measurement unit)*, insieme riescono a fornire consapevolezza della posizione al veicolo autonomo con un rate di aggiornamento di solito pari a 200 Hz.
  Per soddisfare i requisiti di real-time, il GPS ha una frequenza di aggiornamento non elevata ed ha il suo punto di forza nella precisione di localizzazione mentre l'IMU riesce grazie alle forze inerziali a tenere traccia dei movimenti con una frequenza maggiore accumulando un errore progressivo che viene appianato dall'aggiornamento GPS; questa tecnica combinata prende il nome di Kalman filtering e garantisce un'accuratezza dell'errore massimo di un metro.

- Rilevamento e mapping di ostacoli: a tal proposito è stato un elaborato un sistema chiamato LIDAR *(Light Detection and Ranging)* che permette di determinare la distanza di un oggetto o di una superficie utilizzando particolari impulsi laser a lunghezze d'onda differenti rispetto ai normali RADAR, cosicché oltre alla distanza si riesce ad apprendere la densità molecolare dell'ostacolo ed i contorni delle forme percepite nell'ambiente. Un ulteriore strumento a disposizione del task di riconoscimento degli oggetti sono le fotocamere. Ogni veicolo è

solitamente equipaggiato con 8 sensori di questa tipologia che producono dati per 1.8 GB al secondo. Dati meno grezzi ma infinitamente preziosi per la sicurezza generale dell'abitacolo e dei passeggeri sono quelli forniti da SONAR e RADAR, dispositivi meno accurati di altri ma in grado di aiutare i sistemi di sicurezza nel momento in cui esistano pericoli imminenti. I dati in output dei dispositivi di sensing sono sfruttati per essere accettati in input da sistemi di apprendimento basati su reti neurali. In particolare vengono sfruttate delle reti neurali di tipo convoluzionale (CNN) caratterizzate dal fatto che riescono ad elaborare output di dimensioni elevate attraverso una operazione detta appunto di convoluzione che riduce il numero di parametri liberi semplificando l'elaborazione degli strati più interni della rete neurale. Proprio il training globale che si fa di queste strutture di intelligenza artificiale motiva l'utilizzo e l'indispensabilità delle tecnologie cloud computing. I meccanismi per evitare la collisione con ostacoli sono sostanzialmente di due tipi: il primo livello è di tipo proattivo e utilizza i risultati del modulo che si occupa della tracciatura dei movimenti e della predizione del traffico emettendo in output informazioni come il tempo mancante alla collisione o la distanza minima tra due corpi in movimento, il secondo livello invece si occupa di sfruttare i dati del primo livello applicando delle variazioni locali al percorso in maniera del tutto reattiva, cioè scavalcando il primo livello all'interno del processo decisionale, se i dati reali ricevuti dai sensori di movimenti installati sul veicolo mostrano errori nel modello predittivo.

- Tracciatura dei movimenti: le tecnologie di tracking di movimento degli oggetti vengono utilizzate per rilevare gli spostamenti dei veicoli nelle vicinanze e dei pedoni a lato della strada al fine di evitare collisioni. Lo step tecnologico che permette di ottenere risultati soddisfacenti sia sul risultato finale sia sul rispetto dei vincoli di real-time per garantire la sicurezza delle persone e delle cose è avvenuto quando le tecniche di riconoscimento sono state perfezionate sempre con modelli di intelligenza artificiale e Deep Neural Network che possono essere addestrate offline con modelli ad-hoc che danno la capacità di predizione delle azioni successive. Il ragionamento del modulo di *decision planning* è del tutto paragonabile a quello che ogni guidatore fa quando è immerso nel contesto dinamico del traffico dove la sua stra-

tegia di guida è influenzata dalle possibili azioni degli altri guidatori. Per predire le azioni degli altri agenti autonomi che sono responsabili del movimento dei veicoli circostanti, vengono creati dei modelli stocastici a partire dalle posizioni fisicamente raggiungibili e ad ognuna di queste è associata una distribuzione di probabilità che influenza la strategia di guida scelta.

Chiaramente per avere un sistema che rispetti tutti i requisiti di affidabilità richiesti, al fine di poter parlare di veicoli completamente autonomi, i moduli devono essere integrati e devono cooperare insieme in maniera efficacie dando soluzione a vari problemi: i sensori installati sono molteplici ed ognuno di loro lavora ad una frequenza elevata per l'alta dinamicità del contesto e la necessità di dover ridurre al minimo la probabilità che alcuni eventi vadano persi. I dati generati sono di conseguenza di dimensioni non trascurabili in fase di elaborazione dunque un insieme di parti messe insieme al fine di ottenere un sistema complesso locale al veicolo deve considerare il malfunzionamento di una di esse ed infine ogni algoritmo, ogni approccio computazionale adottato deve essere compatibile con i limiti di risorse a disposizione. Dato che ogni parte da sola, se non ben coordinata con le altre, ha un risultato fine a sé stesso ci sono tutti i presupposti per introdurre un sistema operativo denominato *Robotic operating System* (ROS). Un sistema operativo di questa famiglia è organizzato a nodi, ogni task, come per esempio la localizzazione, è ospitato da uno di questi nodi mentre la comunicazione tra le parti è gestita attraverso modelli publish/subcribe oppure a topic come mostrato in figura [1.1].

Di seguito sono invece introdotte alcune delle strategie adottabili da un'architettura software di questo tipo al fine di garantire il rispetto dei requisiti richiesti:

- Affidabilità: i modelli di comunicazione utilizzati fanno riferimento ad un nodo master che non può mai smettere di fornire le proprie funzionalità. Nel momento in cui ci fosse una failure da parte di questo componente, come in tutti i sistemi centralizzati, si avrebbe un tracollo dell'affidabilità, è per questo che ci devono essere meccanismi di elezione di un nodo master di backup.

- Performance: un sistema operativo deve garantire il rispetto di certi target temporali, e la comunicazione tra i vari nodi, qualsiasi sia il

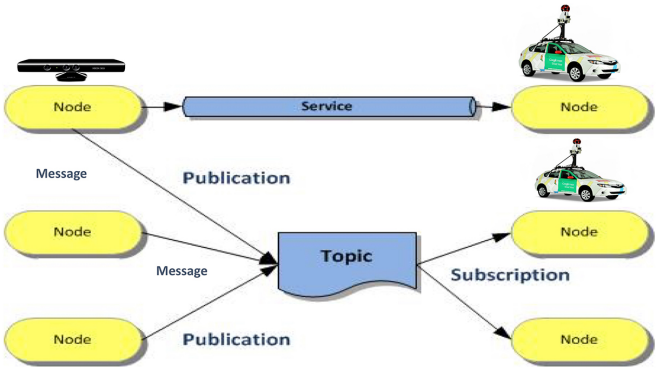

Figura 1.1: Flusso dati all'interno di un ROS.

protocollo utilizzato, introduce degli overhead così come problemi di duplicazione dei messaggi si introducono nella trasmissione in broadcast. Una memoria condivisa con meccanismi di sincronizzazione su tentativi di scrittura concorrente rappresentano una buona soluzione, per lo limitare almeno l'utilizzo di protocolli di comunicazione ai soli moduli di comunicazione esterna (altri veicoli o cloud).

- Sicurezza: immaginando il contesto di deployment dei futuri veicoli autonomi, il problema della sicurezza e dell'integrità di questi sistemi sarà un tema molto caldo. Affidare il controllo dei comandi, la gestione delle situazioni di pericolo e la responsabilità decisionale ad un software presuppone il fatto che una piccola modifica ai dati o al processo decisionale del software stesso possa causare eventi potenzialmente fatali per le persone. È di conseguenza necessario mettere insieme degli standard minimi di sicurezza che proteggano da attacchi attraverso il quale si potrà compromettere il totale funzionamento del veicolo. Esistono molti tipi di attacchi che per esempio possono essere portati anche senza tentare l'intrusione all'interno del sistema stesso; semplici modifiche ai cartelli stradali ingannerebbero i moduli di riconoscimento visivo causando incidenti.

Detto come un software per automobili autonome di livello 5 dovrà a grandi linee essere strutturato, in ultima istanza si valutano quelli che saranno i requisiti hardware per arrivare a garantire certe prestazioni compu-

tazionali.
Al giorno d'oggi a causa del costo dei componenti non si possono rilasciare al pubblico di grande consumo questo tipo di veicoli e la commercializzazione è per questo ancora lontana. Tra le piattaforme sperimentali costruite ad oggi troviamo quella composta da microprocessori e acceleratori grafici del tipo Intel Xeon E5 e Nvidia Tesla K80 i quali hanno costi che si aggirano sui 20.000 dollari per modulo [13].

1.1.2 Infrastrutture per veicoli autonomi

I veicoli autonomi rappresentano gli agenti che si muovono all'interno del traffico urbano ed extraurbano con una capacità decisionale propria.
Per riuscire ad operare necessitano di una serie di miglioramenti infrastrutturali che le comunità nel processo di ammodernamento e sviluppo delle arterie stradali dovranno installare a supporto della circolazione. Il trend che si presuppone sarà seguito ricade all'interno del campo dell'Internet of Things (IoT) e le tecnologie di integrazione con il Cloud Computing. Sono entrambi termini ormai entrati all'interno della conoscenza comune, e si riferiscono, il primo, al concetto di dotare di un'interfaccia ed una connessione in rete degli oggetti di uso comune, mentre il secondo si riferisce al paradigma di erogazione di risorse informatiche su richiesta attraverso la rete. Le città del futuro necessiteranno dell'integrazione del mondo fisico all'interno dell'universo cloud e di conseguenza le infrastrutture intelligenti dislocate all'interno delle città, che saranno introdotte nel prossimo capitolo, dovranno collezionare una quantità elevata di dati che saranno trasmessi alle piattaforme cloud seguendo il cosiddetto *Autonomoous driving stack* [18], uno stack di tecnologie e protocolli del tutto simile a quello utilizzato per l'IoT e di cui altro non rappresenta che una specializzazione detta appunto *Internet of Vehicles (IoV)*. L'IoV può essere definito come una piattaforma che abilita lo scambio di informazioni tra veicoli e l'ambiente circostante attraverso diversi mezzi di comunicazione, come risultato dell'integrazione tra i sistemi di trasporto intelligenti (ITS) e l'IoT. Questa nuova piattaforma arriverà a creare una rete integrata per il supporto di svariate funzionalità (gestione intelligente dei flussi di traffico, fruizione dinamica di informazioni, controllo intelligente dei veicoli ecc ecc..); esso è per lo meno composto da tre diversi componenti fondamentali: la rete inter-veicolare, la rete intra-veicolare e la rete Internet mobile. Il concetto dell'IoV apre le

porte ad una serie di considerazioni ed opportunità per arrivare ad avere vantaggi in termini di costi e benefici per vari attori all'interno del settore della movimentazione delle persone e delle merci. I singoli guidatori non saranno i soli protagonisti di questa rivoluzione, bensì la società intera e le opportunità di business creeranno spazi non esplorati per inserire classi di servizi ad ora non ancora possibili. Questo è testimoniato dalla ingente quantità di investimenti che le grosse società di telecomunicazioni stanno compiendo al riguardo: si stima che il guadagno medio nei costi di gestione annuali di un veicolo possa diminuire di 1.400 dollari da dividere in consumo di carburante, diminuzione di spese per l'individuo e aumento medio della produttività dovuta alla gestione intelligente della congestione stradale [9]. La società avrà vantaggi di minore impatto per ogni veicolo che utilizza le strade pubbliche ma riducendo la possibilità di incidente collegata all'errore umano e l'usura delle infrastrutture nel tempo sarà possibile una migliore allocazione delle risorse pubbliche soprattutto nel reparto della sanità e, aspetto che non deve passare in secondo piano, ci sarebbero miglioramenti nella qualità di vita media delle persone per la minor produzione di anidride carbonica.
Tutto questo come detto creerebbe un'intera nuova categoria di fornitura di servizi in maniera molto simile a quello che è avvenuto con l'avvento degli smartphone e la loro progressiva posizione centrale che si sono guadagnati nella vita quotidiana del loro utilizzatore medio, servizi che saranno legati a segnalatori automatizzati di postazioni di parcheggio, fruizione di algoritmi di navigazione personalizzati oltre che servizi legati alla localizzazione e dedicati alla pubblicizzazione delle attività, servizi di tipo Business-to-Business.

1.2 Internet of Vehicles

Le tecnologie coinvolte all'interno di questa nuova piattaforma sono tante, e l'architettura identificata dalla maggior parte dei ricercatori è composta da tre livelli.

1. Il primo livello contiene tutti i sensori all'interno dei veicoli che raccolgono i dati che sono frutto delle variazioni nel contesto ambientale ed identificano eventi relativi ai dati stessi (triggering events).

2. Il secondo è focalizzato sulla comunicazione e supporta diverse semantiche che differiscono per il tipo di agenti in causa (veicoli, pedoni, biciclette, sensori, infrastrutture).

3. Infine il terzo include servizi ad alto livello realizzati attraverso il cloud computing che includono supporto per il calcolo attraverso moderni approcci quali Big Data.

Questo tipo di architettura soffre di tanti difetti tutti figli dell'approccio accademico alla questione: non vengono infatti considerati tutti quei problemi e tutte le minacce di sicurezza che non possono essere messe in secondo piano da casi d'uso reale; di conseguenza come avviene nella maggior parte dei casi la vera architettura finale è dettata da ciò che sono i requisiti del caso di studio nel mondo reale. Partendo dai tre, già citati layer, bisogna aggiungere una serie di funzionalità implementabili solo colmando le lacune presenti nei layer dell'architettura embrionale a tre strati:

- I meccanismi di sicurezza sono fondamentali e tra questi vanno per lo meno garantiti autenticazione, autorizzazione e comunicazioni sicure ed affidabili.

- Meccanismi di routing intelligente delle comunicazioni con scelta della rete giusta in base alla posizione o alla richiesta del servizio.

- Non esiste un'interazione che non sia quella fruibile attraverso le notifiche mostrate dalle auto con i passeggeri, e questa rappresenta una grossa limitazione nel modello di interattività.

- Manca per lo meno un layer che garantisca il preprocessing delle informazioni raccolte dai sensori: non è per nulla immaginabile una situazione in cui le informazioni vengano incanalate nel traffico di rete senza nessuna attività preliminare, basti pensare ai tanti problemi che emergono dalla sola congestione di rete senza neanche considerare la scarsa qualità delle informazioni.

L'architettura elaborata e studiata da [9], rappresentata nella figura [1.2], è la seguente (i livelli saranno introdotti con un approccio bottom-up):

- **User vehicle interface layer**: è il livello che permette l'interfacciamento dell'utente umano del veicolo con i risultati delle elaborazioni ottenute per via computazionale da tutti i livelli superiori. Gestisce anche la presentazione delle informazioni in base alla tipologia di evento e di notifica utilizzando tutte le possibilità date dagli equipaggiamento di bordo dell'autoveicolo (luci e suoni).

- **Data acquisition layer**: questo livello si occupa delle gestione dei flussi di dati che vengono raccolti nell'ambiente attraverso la suite di sensori di cui un veicolo è dotato ma non solo, fonte di dati sono anche le comunicazioni tra tutti gli agenti intelligenti che vengono coinvolti nell'IoV comprese le unità stradali intelligenti (RSU). Per *RoadSide Unit (RSU* si intendono i dispositivi dislocati lungo le infrastrutture stradali che promuovono la comunicazione wireless effettuata attraverso DSCR *(Direct Short Range Communication)*. Queste forniscono ai veicoli dati ad alta frequenza per segnalare eventi utili alla loro coordinazione.

- **Data filtering and pre-processing layer**: è un livello fondamentale per garantire l'efficienza dell'IoV, esiste infatti una differenza sostanziale tra il concetto di dato e quello di informazione. Il primo è solo una codifica di grandezze misurate e rilevate da un'entità al livello di *Data acquisition* come flussi continui privi di semantica mentre il secondo è il risultato di operazioni di estrazione ed elaborazione compiute a partire dai dati stessi, i quali assumono un significato in relazione ad un determinato contesto. L'insieme di azioni eseguite per arrivare a trasformare i dati in informazione sono appartenenti a questo livello e le preferenze di elaborazione dipendono da profili di personalizzazione dell'utente e del veicolo. È fondamentale che questo tipo di azioni sia eseguito in maniera distribuita affinché il traffico di rete sia costituito da traffico che porta del valore aggiunto all'intera infrastruttura.

- **Communication layer**: risalendo lo stack dei livelli troviamo quello predisposto alla trasmissione delle informazioni che deve garantire la scelta della miglior rete alla quale dover ritrasmettere gli output del livello precedente rispettando i *QoS (Quality of Service) level* specificati dall'utente e dalle esigenze di comunicazione. I livelli di servizio sono utilizzati per indicare i parametri usati per caratterizzare la qua-

lità del servizio offerto dalla rete (ad esempio perdita di pacchetti, ritardo) e vengono accordati per mezzo di contratti denominati *SLA (Service level agreement)*.

- **Control and managment layer**: Dato che ogni rete utilizza metodi di comunicazione differente, protocolli diversi ed in generale si ha una disomogeneità tra i vari provider nei servizi di connessione, un layer che nasconda ai livelli superiori questi problemi è necessario ed indispensabile per permettere alle funzioni ad alto livello di operare utilizzando primitive standard.

- **Processing layer**: È la destinazione dei dati raccolti, essi sono in ultima istanza processati da diversi tipi di infrastrutture cloud sia pubbliche che private. Si raggiunge un livello di qualità delle informazioni ancora più elevato che qui infatti prende il nome di conoscenza, ovvero la capacità di riuscire ad attuare un processo decisionale a partire dalle informazioni stesse abilitando di conseguenza la creazione di tutte le possibilità di business e di *data services*.

- **Security layer** : La sicurezza non può essere considerato un livello all'interno dello stack di IoV come tutti gli altri. In tutte le discipline informatiche la sicurezza è infatti considerata in maniera trasversale e non solo ed esclusivamente nelle interazioni, in ogni singolo momento tutte le operazioni devono avvenire garantendo autenticazione, integrità, *non-repudiation*, controllo di accesso e disponibilità delle risorse. Per non-repudiation si intende la condizione secondo cui non possa essere negata la paternità di un'azione all'interno di un sistema.

La definizione di un'architettura condivisa per IoV promuoverà l'integrazione del settore automobilistico con le più avanzate tecniche di elaborazione delle informazioni, contribuirà allo sviluppo di applicazioni relative all'efficienza energetica dei dispositivi connessi e alla riduzione della mancanza di mezzi di coordinazione e comunicazione tra veicoli. I prodotti, i servizi e la user experience avranno un incremento di qualità elevato dato dai risultati del progresso nello sviluppo di queste tecnologie e della loro penetrazione nel mercato del trasporto urbano privato.

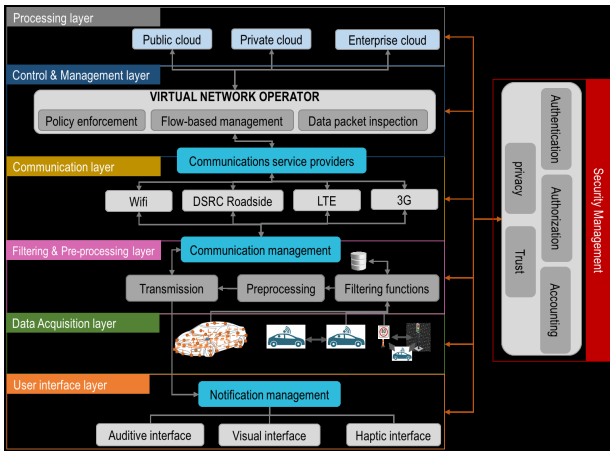

Figura 1.2: Architettura Internet of Vehicles

1.2.1 Standard e protocolli per IoV

La tendenza che caratterizza il futuro dell'evoluzione della rete veicolare ha portato alla necessità di definire un'architettura per una particolare istanza dell'Internet of Things, che riguarda proprio le infrastrutture urbane ed i veicoli autonomi. La caratteristica principale di questo tipo di tecnologie riguarda il valore aggiunto che si crea dallo scambio delle informazioni mentre le difficoltà si presentano quando le entità coinvolte non sono, come nel caso studiato, conosciute a priori, prodotte da compagnie differenti ed appartenenti a epoche tecnologiche differenti. Data la portata dell'investimento che un'automobile comporta infatti i tempi di sviluppo del mercato sono molto più veloci dei tempi medi di ricambio della flotta veicolare. La proprietà fondamentale che, considerati anche gli ostacoli presenti, necessita di essere garantita è **l'interoperabilità**: gli enti ed i consorzi che si impegnano a varare degli standard per tutti i livelli dell'architettura (IETF, IEEE, CEN, W3C) sono già focalizzati sulla definizione delle migliori strategie per permettere ai pionieri del settore di accedere e inter-scambiare i dati che circolano all'interno della IoV.

Le caratteristiche ed i requisiti presentati fanno ricadere la definizione degli standard ed i protocolli all'interno di una particolare configurazione di rete definita come *MANET (Mobile Ad-hoc NETwork)* :

> "Una MANET è un sistema autonomo di router mobili e dei loro host associati, connessi con collegamenti di tipo wireless che sono uniti formando un grafo di forma arbitraria. Tali router sono liberi di muoversi casualmente e di auto organizzarsi arbitrariamente, sebbene la topologia wireless vari rapidamente ed in modo imprevedibile. Tale rete può operare da sola oppure essere connessa alla rete Internet." [22]

Nel caso di studio di questo lavoro, vengono considerate una loro particolare specializzazione che sono le *VANET (Vehicular Ad-hoc NETwork)*, create da veicoli dotati dispositivi di comunicazione a medio raggio che sfruttano la ritrasmissione per raggiungere veicoli a distanze superiori al raggio di copertura oppure dei supporti infrastrutturali statici chiamati *RSU (Road Side Unit)*. Il ruolo dell'IoV e del *vehicular cloud* è quello di estendere i concetti al di sopra delle VANET per aumentare l'enfasi intorno allo scambio delle informazioni tra gli agenti in gioco e guadagnarne in termini di valorizzazione dei dati stessi. Dato che comunque il contesto è quello di un'estensione di una serie di tecnologie già esistenti, e che supportano all'interno di queste reti mobili, la comunicazione intra ed inter-veicolare, di seguito sono elencati alcuni dei protocolli di comunicazione utilizzati:

- Protocolli a livello fisico: le tecnologie utilizzate per la comunicazione a livello fisico vanno dagli infrarossi alle onde radio ed in particolare sono regolate dallo standard IEEE 802.11p WAVE [2], conosciute anche con il nome di Dedicated Short Range Communication (DSRC). Le frequenze che vanno dai 5.855 MHz ai 5.875 MHz sono a disposizione delle applicazioni di consumo e di business, quindi in generale, a tutto ciò che non riguarda la gestione della sicurezza e del traffico per i quali invece è riservata una banda che va dai 5.875 MHz ai 5.905MHz. I protocolli utilizzati a livello MAC invece devono essere affidabili per quanto riguarda l'estrema dinamicità dei componenti della rete e l'alta condivisione della banda di rete, e tra i più utilizzati troviamo il CSMA definito nelle sue caratteristiche dallo standard IEEE 802.3.

- Protocolli di routing: per questo tipo di applicazioni di rete, il protocollo a livello di routing è solitamente un elemento essenziale perché i nodi di rete, data la loro alta dinamicità, comportano una difficoltà maggiore nel mantenere aggiornate le strutture dati collegate agli algoritmi stessi. Tra le varie tipologie presenti trovano colloca-

zione vantaggiosa quelli appartenenti alle macro categorie denominate *transmission strategy* ed *information required* dove i primi hanno l'obiettivo di trasmettere l'informazione da un nodo mittente ad un nodo destinatario in base a strategie che dipendono dal tipo di informazione che un nodo desidera ricevere o da alcune proprietà che lo caratterizzano, mentre i secondi mantengono delle strutture simili a delle tabelle di routing che raggruppano i destinatari in base alla necessità che hanno di ricevere dei messaggi di predeterminate tipologie. A prescindere dall'efficienza che i protocolli di routing possono avere all'interno di questo particolare scenario di comunicazione, la sfida principale include il superamento di ostacoli quali un accesso alle frequenze radio eterogeneo, problemi di connessione in ambienti non solitamente abitati (si pensi ai problemi di connessione degli smartphone attraversando alcune località in automobile), la perdita dei pacchetti, l'aggiornamento proattivo delle connessione e delle tabelle di routing. Sono tematiche molto importanti, per le quali non sono state trovate soluzioni all'interno delle VANET, un'utopia nel momento in cui non si garantisce affidabilità ai protocolli di rete anche nel caso in cui ci siano problemi di gestione delle situazioni critiche.

Stiamo presentando i requisiti e le plausibili soluzioni tecnologiche già presenti da cui poter evolvere fino ad ottenere una suite di protocolli che riescano a realizzare l'IoV, l'ambiente intelligente in cui possano immergersi i veicoli a completa guida autonoma e dove andremo a studiare le strategie di coordinazione per evitare collisioni ed ottimizzare il loro comportamento alle intersezioni.
Prima di parlare della coordinazione stessa e degli algoritmi che la regolano in particolare manca un requisito fondamentale: la sicurezza. L'integrazione di differenti tecnologie, servizi e standard prodotti e sviluppati da costruttori estremamente eterogenei richiede una sicurezza estrema dei dati per ragioni che spaziano dalla privacy personale alla sicurezza dei cittadini all'interno degli abitacoli e dei pedoni che popolano quotidianamente le strade urbane; la sicurezza informatica è una materia molto delicata e che parte dallo studio delle vulnerabilità del sistema considerato. L'IoV considerato nel suo insieme presenta una miriade di vulnerabilità estremamente variegata che vanno dalla difesa fisica dei dispositivi (in particolare è difficile garantire la sicurezza fisica di un autoveicolo o di un dispositivo come un RSU) al controllo delle informazioni che fluiscono attraverso i nodi di

rete passando per l'estrema eterogeneità dei software (numero di vendor estremamente diversificato con regole diverse per nazioni diverse) e la difficoltà nella certificazione o addirittura l'imposizione di centraline unificate che sembrano ad oggi un passo ancora troppo difficile da compiere. Una volta che un cybercriminale entra in possesso del flusso di dati che regolano il funzionamento di un agente autonomo può indurne il comportamento in maniera molto agevole e manipolarne componenti differenti come freni, chiusure di sicurezza e airbag come già dimostrato durante una conferenza sulla cybersecurity su un modello di Jeep Cherokee [7].
I trattati sulla sicurezza per l'IoV propongono soluzioni sia per migliorare la protezione delle infrastrutture critiche sia per fornire uno schema di comunicazione all'interno di applicazioni VANET fornendo i dettami per uno schema di comunicazione che rispetti i requisiti di sicurezza che sono i seguenti:

- *Data authentication*: qualsiasi informazione che viene trasferita deve essere riconducibile all'identità del veicolo che la produce.
- *Data integrity*: i dati trasferiti e ricevuti in ogni comunicazione devono essere consegnati correttamente.
- *Data confidentiality*: i dati devono essere protetti al fine di assicurare una trasmissione segreta end-to-end.
- *Access-control*: ogni singola entità che è coinvolta nell'IoV deve essere abilitata all'accesso dei servizi per cui detiene i privilegi.
- *Data non-repudiation*: nessun veicolo può negare l'identità di un altro veicolo.
- *Availability*: la comunicazione tra i veicoli deve essere disponibile a prescindere dalle condizioni differenti che l'ambiente offre.
- *Anti-jamming*: deve essere sempre presente un meccanismo di prevenzione che impedisca interferenze maliziose e volontarie nella comunicazione tra veicoli.

Devono essere adottate delle contromisure che aderiscano ai requisiti sopra definiti, ma data la dimensione dell'IoV non è un compito semplice andare ad analizzare attraverso una tecnica ben precisa la totalità delle possibili

minacce: un tool molto popolare per la modellazione delle minacce è *Microsoft STRIDE* basato sulla costruzione di grafi per lo studio dei rapporti di causa ed effetto sull'IoV, i grafi e la teoria ad essi collegati degradano nelle prestazioni una volta applicati in ambito industriale. Per questo la modellazione avviene attraverso un approccio matematico: l'intero IoV viene definito come un sistema lineare che varia nel tempo, mentre gli attacchi portati alla sicurezza dell'integrità del sistema, l'iniezione di input malevoli oppure minacce alla disponibilità di servizio sono modellati come fonti di disturbo. Questi sistemi lineari per garantire il rispetto dei requisiti di sicurezza forniscono delle invarianti di sistema su cui nascono tutte le tecniche di difesa a partire dalle prime e fondamentali che sono le tecniche di prevenzione.
Se le tecniche di prevenzione falliscono acquistano vitale importante i cosiddetti *Intrusion detection system (IDS)*, i quali sono sistemi di protezione attiva che, analizzando la rete, vanno alla ricerca di comportamenti potenzialmente dannosi alle policy di sicurezza definite per il sistema stesso; tutto questo avviene con delle tecniche che combinano l'analisi con una sorveglianza mirata verso parti critiche del sistema chiamate *honeyspot* che, come il nome suggerisce, servono ad ingolosire possibili attaccanti nel tentativo di spostare l'attenzione dalle vere risorse vitali del sistema. Sono comunque approcci che considerano tutti processi attivi e che impiegano risorse che non possono essere destinate a tenere alta l'efficienza dell'IoV, e le loro funzioni sono disabilitate in situazioni giudicate sicure nel ciclo di funzionamento del sistema.
Oltre alla prevenzione e alla rivelazione di minacce ed attacchi hanno un ruolo fondamentale anche i protocolli di routing che garantiscono l'integrità, la confidenzialità e l'anti-jamming nelle comunicazioni tra i veicoli così da evitare re-routing dolosi delle comunicazioni da una parte e la modifica del payload del pacchetto trasmesso dall'altra: le tecniche più comunemente utilizzate sono le funzioni hash, esse sono in grado di fornire una soluzione a tutte le criticità di sicurezza che si presentano nelle comunicazioni e nei vari hop che ogni pacchetto fa tra i router che sono presenti nel'intermezzo della comunicazione, tra i più utilizzati si possono citare SRP e SOADV [14].

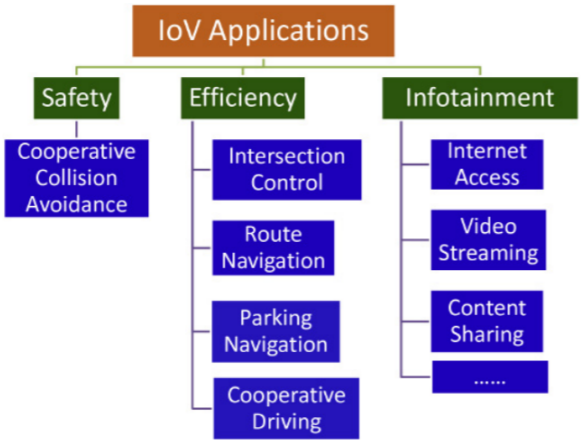

Figura 1.3: Applicazioni IoV

1.3 Applicazioni e sviluppi per IoV

Le applicazioni di IoV sono svariate, e si possono categorizzare all'interno di tre grandi aree seguendo la tassonomia in figura 1.3 [**?**].

La comunicazione tra veicoli e la possibilità di accedere alle informazioni in tempo reale prodotte da chi si trova nelle immediate vicinanze migliorano il livello medio di sicurezza dei trasporti. Grazie a un insieme di sistemi di prevenzione delle collisioni, messi in funzione dalla cooperazione dei sistemi automatici installati nei moduli computazionali dei veicoli, dei sistemi di segnalazione del pericolo possono funzionare anche su veicoli con conducente e con caratteristiche di autonomia non elevate. Di pari passo con la sicurezza viaggiano le possibilità che IoV apre al mondo dell'efficienza dei trasporti: sfruttare le informazioni prodotte in termini di ottimizzazione del controllo alle intersezioni, scelta del percorso di navigazione e cooperazione nella distribuzione del carico sulle arterie di circolazione del traffico determinano un miglioramento dei consumi energetici e dei tempi di circolazione delle merci con tutti i vantaggi in termini economici, e non solo, che ne conseguono. Il calcolo del percorso, in uno scenario dove i veicoli sono connessi, diventa un problema di ottimizzazione del tempo e dei consumi dove i vincoli sono posti in tempo reale dai dati sul traffico collezionati dalle RSU. Il cooperative driving si riferisce invece alla coordinazione delle code

di veicoli per fare in modo che essi guidino come se fossero uno solo, qui si aumenta l'efficienza attraverso il cosiddetto effetto **platooning** che ricalca il movimento delle carrozze di un treno dove la distanza tra due carrozze seguenti rimane sempre la stessa. Nel normale traffico veicolare questo non può essere ottenuto perché i tempi di reazione di ogni guidatore in fase di accelerazione e decelerazione decrementano le prestazioni di efficienza delle code in maniera drammatica. Il terzo ramo applicativo importante per l'IoV riguarda l'intrattenimento: la connessione dei veicoli determina anche l'accesso garantito alla rete Internet e a tutti i suoi servizi e vantaggi. La creatività delle idee e la qualità delle applicazioni che si creeranno in futuro garantiranno guadagni alle compagnie produttrici ed un'esperienza di viaggio molto più confortevole ai passeggeri dei veicoli. Essendo IoV un'estensione delle reti VANET, le quali non erano strutturalmente adatte per gestire l'enorme mole di dati prodotta a flusso costante dai veicoli, sta ancora oggi muovendo i primi passi e le ricerche effettuate producono risultati validi in linea teorica che non hanno riscontri nel mondo reale per una serie di problemi che si potranno risolvere solo quando il passaggio ai veicoli connessi sarà largamente accettato ed in funzione. Le seguenti sono alcune tematiche che gli studi futuri dovranno prendere in considerazione per aumentare affidabilità ed efficienza dei meccanismi che contraddistinguono l'IoV:

- Routing efficiente delle informazioni: nonostante siano stati sviluppati protocolli di routing ad-hoc per le reti VANET, la grande mobilità della topologia di rete e la possibilità di connessioni non sempre stabili rappresentano un ostacolo non banale per la diffusione delle informazioni tra i veicoli.

- Meccanismi di coordinazione per veicoli: la coordinazione tra i veicoli non è un problema banale. Il controllo delle intersezioni ed in generale la sincronizzazione nelle applicazioni distribuite deve essere sviluppata in maniera disaccoppiata dalla logica applicativa. Nel seguito saranno studiati e trattati in termini di simulazione alcuni approcci di coordinazione per veicoli a guida autonoma, ma non tutti gli aspetti del mondo reale (pedoni, veicoli d'emergenza ecc.) sono catturati ed i risultati ottenuti sono infatti solo sperimentali.

- Elaborazione dei flussi di dati: seppur esistano molte tecniche che riguardano i Big Data già utilizzate in tanti settori, bisogna considerare lo sviluppo di tecniche ad-hoc pensate per i flussi di dati cloud-based.

Capitolo 2

Traffico e coordinazione veicolare

La densità dei veicoli in circolazione sulle arterie stradali è notevolmente aumentata negli anni a causa della facilità di accesso a mezzi di trasporto privato da parte della quasi totalità dei cittadini, diventando un problema affrontato sia dalla matematica che dall'ingegneria. In particolare il flusso del traffico è lo studio delle interazioni tra i viaggiatori (pedoni, ciclisti, veicoli ecc. ecc.) e le infrastrutture (autostrade, segnali e dispositivi di controllo del flusso) che si pone l'obiettivo di comprendere e modellare una rete di trasporto efficiente che riduca il problema della congestione.
Il traffico si comporta in maniera complessa, non lineare e dipende dalle singole interazioni tra gli attori in gioco. Molti modelli matematici riassumono i fenomeni che si manifestano da queste interazioni attraverso la formazione di cluster e la propagazione di tipo *shock wave*.

In particolare l'ingegneria stradale utilizza gli studi effettuati nel campo della fluidodinamica al fine di minimizzare la congestione stradale, cioè:

> "...la condizione relativa ad una rete il cui utilizzo aumenta progressivamente fino a pervenire a situazione di bassa velocità, lunghi tempi di viaggio ed incremento delle code. Questo accade quando il traffico (cioè il flusso veicolare) è superiore alla capacità della strada (o a quella di una intersezione lungo di essa)."

Le opere architetturali sono importanti e fondamentali per prevenire il fenomeno citato ma a causa delle difficoltà che si presentano nel rinnovamento infrastrutturale, hanno un ruolo di primo piano le tecniche di coordinazione per i veicoli in accesso alle intersezioni. Sono infatti le intersezioni e le immissioni agli svincoli nelle strade a lunga percorrenza le due principali cause di rallentamento dei flussi di traffico.

2.1 Flussi di traffico

Lo studio dei flussi di traffico è generalmente sottoposto ad una modellazione bi-dimensionale che riassume le variazioni nello spazio e nel tempo e si presta molto bene ad una comprensione grafica dei risultati su diagrammi cartesiani che evidenziano le situazioni di congestione. Le tre variabili principali in gioco, le cui variazioni permettono di determinare le soglie di congestione per le arterie stradali considerate, sono le seguenti:

- *Velocità* (v) : intesa come grandezza lineare che descrive la variazione di spazio percorso nell'unità temporale. Nello studio della teoria del traffico sono importanti due misure diverse di velocità: la prima detta *"velocità media nel tempo"* e la seconda *"velocità media nello spazio"*. Le due grandezze sono semanticamente simili perché considerano entrambe una misura media riassuntiva ma diverse nel modo in cui la velocità viene campionata, nel primo caso si utilizzano dei *detector loops* che misurano la velocità istantanea di passaggio in punti prefissati delle strade stesse mentre nel secondo caso si analizza la velocità media di crociera dei veicoli nello spazio considerato. fig.[2.1]

- *Densità* (k) : indica il numero di veicoli presenti nell'unità di spazio. È una misura importante per quantificare in uno specifico caso di studio la soglia di densità critica, definita come la massima densità raggiungibile in condizioni di traffico libero, e la soglia della cosiddetta *jam density* riferita invece alla situazione di congestione. Solitamente vale la regola che la soglia massima di densità in condizioni di congestione sia sette volte la soglia massima in condizioni di traffico libero.

- *Flusso* (q) : indica il numero di veicoli per unità di tempo che passano un punto di riferimento preciso. È l'inverso del tempo medio che intercorre tra il passaggio di due veicoli.

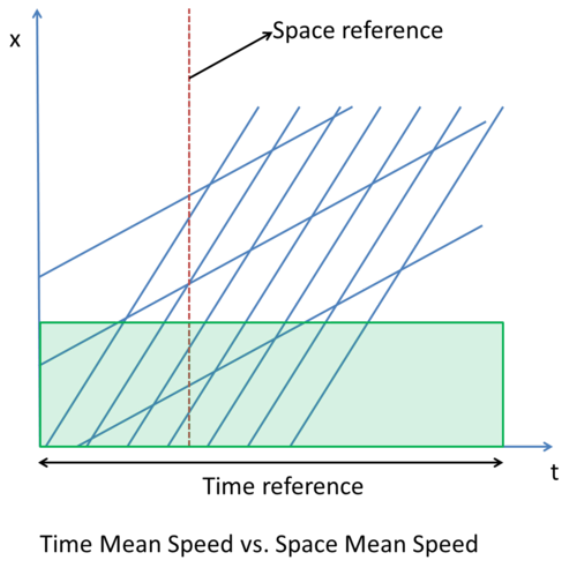

Figura 2.1: Grafico velocità

Considerando una singola traiettoria stradale, potremmo facilmente costruire un modello che definisce, fissati i vincoli sulle variabili, quali sono le grandezze massime in gioco per i flussi di traffico prima di arrivare al verificarsi di fenomeni di congestione.
Ciò che complica lo studio dei flussi di traffico sono le interazioni nei punti di conflitto delle traiettorie: le **intersezioni**.
Più flussi confluenti da strade con capacità elevate generano alle intersezioni dei veri e propri colli di bottiglia per i flussi stessi con cause imputabili in maniera chiara e netta soprattutto a due fattori:

- **Riduzione delle velocità dei flussi in transito**, un comportamento che si rende necessario per permettere ai conducenti di valutare il contesto e, in base alle regole vigenti all'intersezione stessa, decidere se proseguire o completare l'arresto.

- **Ripartizione del tempo di via libera**, esistono limiti fisici tali per cui l'accesso ad una risorsa condivisa non può essere concesso per la totalità del tempo a tutti i flussi entranti. La divisione temporale limita il deflusso delle diverse correnti.

Le intersezioni sono a tutti gli effetti delle risorse condivise tra i flussi concorrenti in entrata, esattamente come lo è la capacità computazionale tra processi in un sistema di calcolo. L'obiettivo è la ricerca di soluzioni che massimizzino il deflusso delle correnti in entrata alle intersezioni, del tutto equivalente alla necessità di mantenere una variazione del flusso in uscita non troppo negativa rispetto alla misura del flusso in entrata.
Storicamente sono state adottate convenzioni e regolamentazioni diverse sia per l'obiettivo prefissato di ottimizzazione dei flussi sia per garantire standard minimi di sicurezza agli utenti della strada.
Le categorie di interventi possibili sono sostanzialmente due [4]:

- **Distribuzione temporale dei flussi**: si concede l'area di intersezione tra i flussi ad uno o più di loro contemporaneamente secondo una regolazione a precedenza (detta anche ad indice di priorità) o una regolazione semaforica.
- **Distribuzione spaziale dei flussi**: si tratta di interventi fisici alla conformazione delle arterie stradali che evitano ai flussi più consistenti di entrare in conflitto tra di loro. Si realizzano corsie dedicate alla svolta, rotatorie, sovrappassi e svincoli su più livelli.

Per la definizione di modelli per la gestione dei flussi di traffico veicolare esiste una terminologia standard, i termini più importanti sono riassunti nella schematizzazione in figura [2.2] :

- *Ramo*: sono gli afflussi in entrata ed in uscita ad un'intersezione, ognuno può avere una o più *corsie*.
- *Corrente*: è costituita dai veicoli che provengono dallo stesso accesso (corsie di un ramo in ingresso all'intersezione) ed è formata da una o più manovre.
- *Manovra*: rappresenta uno dei possibili movimenti di attraversamento di un'intersezione e permette lo spostamento di un veicolo dal ramo d'accesso a quello di uscita.

Al fine di sviluppare modelli di accesso alle intersezioni efficienti è fondamentale massimizzare la condivisione nell'utilizzo da parte delle correnti la cui compresenza a livello spaziale o temporale non rappresenta un pericolo per la sicurezza (cioè correnti le cui possibili manovre non si sovrappongano

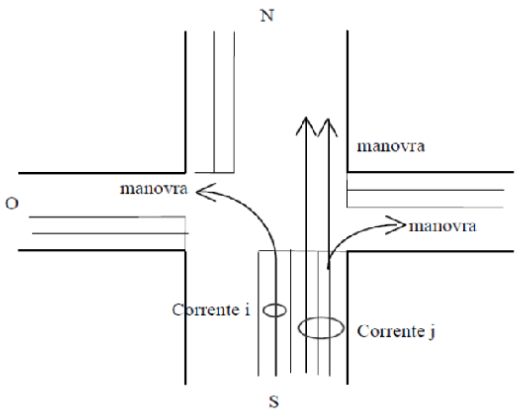

Figura 2.2: Componenti di una intersezione

nello spazio causando collisioni), si definiscono a questo proposito due tipi di relazione tra le correnti in ingresso ad un'intersezione:

- *Correnti compatibili*: si definiscono correnti compatibili due correnti che permettono di attraversare l'intersezione in sicurezza.
- *Correnti incompatibili* (per attraversamento o confluenza): si definiscono correnti incompatibili due correnti che non permettono l'attraversamento contemporaneo dell'intersezione in sicurezza, poiché una o più manovre corrispondenti generano dei punti di conflitto lungo le traiettorie che delineano. Questa tipologia di correnti sono quelle che generano concorrenza sull'utilizzo della risorsa condivisa.

2.1.1 Intersezioni a regolazione semaforica

La tecnica di regolazione dei flussi basata sugli apparecchi semaforici è quella tradizionalmente più utilizzata per tutti gli incroci caratterizzati da flussi rilevanti, cioè dove una semplice regolazione basata sulle precedenze non garantisce sicurezza ed efficienza del deflusso.
Storicamente sono stati eseguiti esperimenti fin dai primi anni del '900, ancor prima che esistessero i microcontrollori, gli impianti erano azionati e regolati nel loro funzionamento da microfoni stimolati dai suoni dei clacson

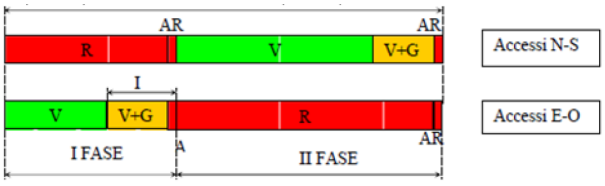

Figura 2.3: Fasi semaforiche

delle autovetture in arrivo. Ora, ove non sono stati sostituiti da regolazioni a rotatoria, utilizzano cicli di funzionamento evoluti e basati su simulazioni artificiali effettuate con elaboratori elettronici. I semafori sono una tecnica che, se non ben analizzata, può sembrare banale e scontata oltre che poco efficiente per evitare o mitigare il formarsi di fenomeni di congestione.
Proprio da un'introduzione sui differenti schemi implementabili da insiemi di semafori si arriva a comprendere il ruolo fondamentale della coordinazione per l'ottimizzazione dei flussi. L'insieme delle primitive a disposizione di questi segnali a percezioni visiva si limita al riconoscimento cromatico del segnale stesso, è lo studio fatto proprio attraverso la modellazione matematica del sistema globale che riesce ad ottenere una parametrizzazione soddisfacente per il problema analizzato.
L'evoluzione in chiave moderna delle primitive di comunicazione tra veicoli porta una capacità espressiva talmente elevata messa a disposizione dei progettisti paragonabile ad un vero e proprio cambio di paradigma. L'IoV e le tecnologie cloud saranno infatti in grado di inquadrare la modellazione del dominio di sistema all'interno dei sistemi multi-agente con tutti i vantaggi all'interno della dimensione delle iterazioni che questo comporta. Il susseguirsi delle ben note luci di colore verde, giallo e rosso nell'ordine, è detto **ciclo**, la cui durata è data dalla somma della durata delle tre *fasi* (una per ogni luce). Una fase è una frazione del ciclo durante la quale il segnale rimane fisso per ognuno dei dispositivi di un gruppo, cioè l'insieme dei semafori che insieme collaborano per regolare il deflusso ad un'intersezione. [2.3].

In questo ambito di regolazione di accesso alle intersezioni banalmente la saturazione avviene quando il numero di veicoli che chiede di passare è maggiore del numero di veicoli che ogni singolo ciclo riesce a servire. La portata segue in maniera ciclica l'andamento della fase, aumentando fino al

massimo passati pochi istanti dopo l'accensione della luce verde, e facendo l'inverso dalla comparsa del giallo fino al segnale di colore rosso. Il flusso, detto anche *portata di saturazione* per una singola intersezione è pari al massimo flusso di smaltimento dei veicoli durante i periodi di via libera e con accesso saturo.
Nelle condizioni di accesso saturo si consideri il deflusso dei veicoli per una corsia: l'inizio della fase di segnale verde coincide con l'attivazione del processo di reazione del conducente che si trova in fronte alla coda, che riparte facendo trascorrere un certo tempo prima del passaggio. Per ogni veicolo seguente la velocità crescerà in maniera graduale fino ad avere una velocità di passaggio pari a quella di regime diminuendo il tempo che separa il passaggio di due veicoli seguenti. Il fenomeno che si ottiene a causa delle fasi semaforiche è la dispersione del plotone veicolare descritto dal cosiddetto modello TRANSYT [17], un'equazione differenziale che riassume i rapporti tra i flussi in maniera discreta in arrivo alla k-sima intersezione. Tale equazione rappresenta il fondamento per lo studio e l'applicazione di metodi di ottimizzazione dei flussi basati su impianti semaforici successivi, il cui più famoso esempio è la famosa *onda verde*. Non esistono, se non in una configurazione dinamica dei parametri che regolano le fasi dei cicli, possibilità di ulteriore miglioramento dell'efficienza semaforica. La più grande limitazione quindi di questo approccio è data dai flussi dinamici nel tempo e dalla parametrizzazione di apparecchi che determinano a priori la gestione della congestione.
La modernizzazione ha permesso la realizzazione di una seconda categoria di semafori caratterizzati dalla capacità di adattare le fasi ottimizzando i deflussi in base alle correnti in ingresso rilevate da reti di sensori e che prendono il nome di *impianto a regolazione attuata*, differenziandosi dagli impianti tradizionali denominati *impianti con funzionamento a piani definiti*. Le reti di sensori utilizzati per i semafori a regolazione attuta sono della famiglia dei dispositivi denominati *Road Side Unit* all'interno del contesto dell'Internet of Vehicles, ma non sono in grado di cambiare l'ordine di grandezza delle prestazioni dei modelli di coordinazione: solo l'effettiva connessione in rete dei veicoli e la comunicazione Vehicle-to-Vehicle o Vehicle-to-Infrastructure apriranno nuove possibilità alla coordinazione che ne trarrà giovamento in termini sia di prestazioni che di sicurezza.

2.1.2 Regolazioni a rotatoria

Nei modelli di coordinazione dei flussi di traffico non a guida autonoma hanno recentemente guadagnato grande considerazione le soluzioni a rotatoria soprattutto grazie a tre motivi:

1. aumento della capacità di intersezione;
2. miglioramente della sicurezza;
3. aumento del campo di applicabilità del tipo di intersezione.

Il miglioramento della gestione dei flussi in entrata è dovuto sia alla precedenza data ai veicoli che percorrono l'anello sia all'assenza di un segnale simile a quello di colore rosso di un impianto semaforico che blocca a prescindere una corrente entrante. La precedenza all'interno dell'anello è lo stratagemma che impedisce l'autosaturazione. La sicurezza è garantita dalla riduzione dei punti di conflitto [2.4] , la moderazione della velocità in arrivo e l'assenza di una grande disparità di velocità in caso di urti tra veicoli.

È importante allo stesso tempo sottolineare gli svantaggi che questo tipo di approccio ha: innanzitutto i costi di realizzazione sono ingenti, e l'impatto sull'ambiente non indifferente; a questo si aggiunge la difficoltà nella coesistenza di ciclisti, pedoni e veicoli nell'attraversamento dell'anello rotatorio.

Calcolare la capacità di deflusso di una rotatoria non è un problema facilmente affrontabile perché ci si trova di fronte ad una tipologia di deflusso che non dipende dalla coordinazione di un elemento centrale ma da scelte che vengono compiute da ogni singolo automobilista. Risulta fondamentale in tal senso la valutazione soggettiva del *distanziamento temporale* che esiste tra due veicoli della corrente principale circolanti sull'anello; in questo senso l'accesso all'intersezione dipende dalle condizioni di congestione dell'intersezione stessa e al di sopra di una certa soglia che caratterizza le correnti in accesso esiste il rischio di deadlock o starvation. Se si blocca il traffico circolatorio a causa di cattive valutazioni del distanziamento temporale, e più veicoli tentano allo stesso momento un'immissione che risulterà inappropriata, allora il flusso circolatorio si bloccherà. Non esiste infine un meccanismo che impedisca ad una singola corrente in accesso costituita da

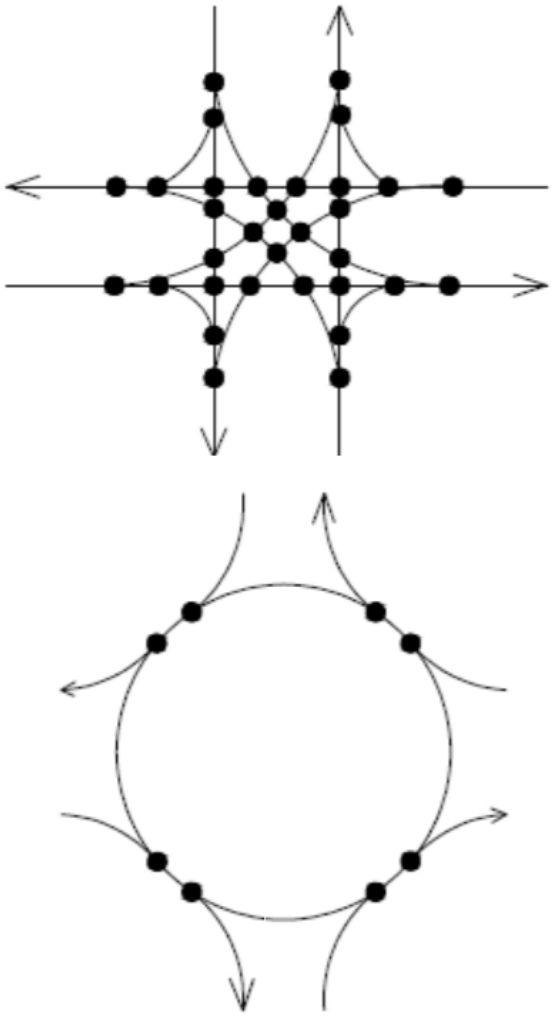

Figura 2.4: Zone di conflitto a confronto.

un flusso più elevato rispetto agli altri di monopolizzare l'accesso al flusso circolatorio, e di costituire in particolari configurazioni di manovre in uscita l'unico a riuscire ad avere un distanziamento temporale valido per l'immissione di altri veicoli sempre provenienti dal flusso stesso, rendendo per le altre correnti inaccessibile l'immissione alla rotatoria. Esistono tecniche di progettazione che studiando a priori le statistiche delle manovre ed evitano il configurarsi di tali scenari ma non li rendono di fatto impossibili.

2.2 Coordinazione di veicoli connessi ed autonomi

Per quanto l'installazione di regolazioni a rotatoria abbia risolto parte dei problemi evidenziati dall'approccio a semafori, le limitazioni nella coordinazione di plotoni di veicoli non completamente connessi non permettono di abbattere al di sotto di una certa soglia i costi diretti ed indiretti che gli utenti della strada pagano.

L'innovazione e le tecnologie abilitanti introdotte nel capitolo 1 hanno mostrato come il trend di sviluppo porterà in tempi non così lunghi all'implementazione di auto a guida autonoma. La coordinazione di veicoli a guida autonoma e l'approccio all'attraversamento delle intersezioni sono un tema fondamentale per la realizzazione del sistema complesso che deriva dalle interazioni dei singoli agenti autonomi. Il sistema complesso e caotico che ne deriverà rappresenterà in questo caso l'intera porzione di traffico veicolare urbano e, escludendo la conduzione della tratta lungo le corsie e gli accessi alle intersezioni, si apre un mondo tutto nuovo per la contesa del diritto di accesso all'attraversamento. Con le capacità di valutazione del contesto ed una precisione nel controllo sempre crescenti, la cooperazione nella gestione delle intersezioni sarà realizzata non solo dai veicoli ma dalle intersezioni stesse. Le caratteristiche di autonomia degli agenti in gioco e le comunicazioni rese possibili attraverso l'infrastruttura di Internet Of Vehicles presuppongono un framework di riferimento del tutto aderente ai principi di un *Multi Agent System* (MAS).

In un MAS un ruolo fondamentale è la dimensione dell'interazione, del tutto indipendente da quella computazionale. Gli agenti infatti gestiscono il loro flusso di controllo in maniera del tutto autonoma ma nessun tipo

di assunzione può essere fatta né sul ciclo di vita né sulle azioni che un agente farà perché del tutto dipendenti da ciò che a livello locale in maniera autonoma decide. È qui allora che le interazioni assumono un ruolo fondamentale come comportamento osservabile di un'entità; si crea così uno *spazio delle interazioni* in cui inserire dei meccanismi di governo. Questi meccanismi che vanno a regolare il comportamento degli agenti prendono il nome di *modelli di coordinazione*.

Un **modello di coordinazione** è ciò che tiene insieme, attraverso la dimensione delle interazioni, tante attività separate formando una comunità d'intenti.

Una nuova modellazione basata su sistemi multi-agente non è solo un'occasione di miglioramento delle prestazioni e dell'efficienza ma anche una necessità data da requisiti che mutano ed elementi del dominio nuovi che delineano vincoli diversi e di conseguenza portano al bisogno elementi di coordinazione ripensati. L'ambiente e la percezione del contesto dinamico in cui si muovono le automobili autonome sono due requisiti che devono essere al centro della soluzione considerata, è anche per questo motivo che ripensare l'intero sistema traffico come un MAS che promuove l'abilità sociale e la *situatedness* degli agenti risulta quella migliore.
Di seguito sono riportati i requisiti minimi e fondamentali su cui un modello di coordinazione per un sistema multi agente di questo genere si deve fondare:

- *Autonomia dei componenti*: L'autonomia è alla base della decentralizzazione del controllo. Se l'intero meccanismo di coordinazione fosse centralizzato avere dei modelli efficienti e sicuri sarebbe un'utopia a causa delle innumerevoli limitazioni che ogni sistema centralizzato comporta: unico punto di rottura, richieste di risorse computazionali enormi ecc. ecc.

- *Comunicazione semplice*: Mantenere il numero dei messaggi contenuto e la quantità di informazioni scambiate al minimo comporta grossi vantaggi per il sistema nella sua globalità. Vengono infatti promosse sia la scalabilità che l'affidabilità. Questo requisito deve essere valutato insieme all'autonomia dei componenti: una completa decentraliz-

zazione della coordinazione comporta infatti un aumento esponenziale del numero di interazioni tra le parti.

- *Standardizzazione dei protocolli*: Senza una standardizzazione dei meccanismi di comunicazione, ogni agente sarebbe costretto a comprendere sia la sintassi che la semantica della comunicazione con ogni altro elemento del sistema. La realizzazione e l'introduzione di nuovi veicoli sarebbe quindi resa possibile a tutti i produttori privati di cui venga certificata l'aderenza ai protocolli standard.

- *Deadlock/Starvation*: deadlock e starvation sono situazioni che non si devono mai verificare per nessun veicolo all'interno del sistema in attesa di ottenere accesso un'intersezione. Ogni veicolo entrante deve vedere accolta la propria richiesta senza rimanere in un'attesa indefinita, anche a costo di andare alla ricerca di compromessi con altri veicoli che inficiano sulle prestazioni globali.

- *Sicurezza*: in questo frangente si intende la sicurezza fisica dei veicoli, la quale deve essere garantita in ogni intersezione. Vale il principio che se tutti i veicoli seguissero il protocollo stabilito allora nessuno di loro subirà una collisione durante l'attraversamento. Procedure di sicurezza e di recupero da situazioni di malfunzionamento devono ridurre i rischi al solo caso in cui uno o più agenti abbiano grossi malfunzionamenti.

- *Sviluppo incrementale*: Questo sviluppo che tende a cedere il controllo della viabilità ai sistemi computazionali deve essere reso possibile in maniera graduale lungo due dimensioni diverse. La prima delle due riguarda la selezione e l'applicazione di metodi di coordinazione tra agenti autonomi a sole poche selezionate intersezioni, la seconda riguarda invece la necessità di creare modelli di coordinazione compatibili anche con veicoli a guida parzialmente autonoma. Ogni step incrementale dovrà essere messo a punto quando la percentuale di veicoli autonomi sul totale o quando il numero di intersezioni coinvolte saranno sensibilmente aumentati così da riuscire a portare un upgrade nell'efficienza della performance globale del comportamento del sistema.

La modellazione del dominio viene effettuata come già detto aderendo ai principi dei sistemi multi-agente. Per i veicoli a guida autonoma di conseguenza deve essere data per assodata la capacità di essere situati nell'ambiente in cui si trovano ad operare e la dimensione computazionale governa il comportamento di ogni singolo agente secondo le regole che vengono programmate. Per affrontare l'argomento che riguarda l'accesso alle intersezioni aderendo ai requisiti appena fissati bisogna parlare della coordinazione nei MAS. Una caratteristica che è propria dei MAS si riferisce al fatto che ognuno degli agenti coinvolti è dotato di capacità decisionale e di conseguenza risultano avere un ruolo fondamentale anche le intersezioni che diventano ancor più evolute e prendono il nome di *Intelligent Intersections*. Questo tipo di intersezioni innovative rientrano all'interno del processo evolutivo che ci porterà all'effettiva creazione dell'Internet of Vehicles e ne saranno parte integrante e fondamentale. Nella terminologia VANET esse sono una sottoinsieme delle infrastrutture attive e ricoprono il ruolo di interlocutore per gli schemi di comunicazione I2V. Nello specifico ci si concentra solo su questo tipo di infrastruttura, perché utili a comprendere il ruolo fondamentale che avranno nelle future intersezioni dedicate ai veicoli a guida autonoma ed in generale nell'intera società multi-agente appena introdotta.

2.2.1 Intersezioni intelligenti

Le intersezioni intelligenti sono parte fondamentale ed integrante dell'intero sistema che deriva dalla cooperazione dei veicoli autonomi ed in questo hanno un ruolo primario. Le caratteristiche dei sistemi multi-agente fanno sì che ognuna delle parti abbia una visione parziale e limitata al suo ambiente circostante mentre la globalità dello stato del sistema, o di una sua sottoparte, possa essere percepita attraverso la cooperazione e le interazioni tra gli agenti. Nel problema specifico lo scenario presenta variabili ed entità in gioco che sono al di fuori del sistema facendone comunque parte: si pensi a pedoni, ciclisti e al loro comportamento che può essere imprevedibile ed inaffidabile. Nei pressi delle intersezioni nessun sistema di percezione di cui sia dotato un veicolo è in grado di avere un'affidabilità individuale tale per cui una manovra di emergenza possa essere eseguita data la caoticità del luogo. E allora, per lo meno in quegli attraversamenti stradali più delicati si rendono necessarie delle soluzioni che contemplano infrastrutture di coordinazione attiva dette appunto **Intelligent Intersections** [15]. Quando il

numero di corsie di cui è costituito un flusso in accesso ad un'intersezione cresce, cresce in maniera esponenziale il numero di manovre possibili per l'attraversamento: un protocollo completamente decentralizzato per la gestione dei conflitti non rientrerebbe all'interno dei canoni di sicurezza e di efficienza necessari.

Se ogni veicolo che costituisce una corrente in accesso all'intersezione fornisce delle informazioni sul proprio stato allora la capacità decisionale può essere centralizzata, e modelli di coordinazione basati su un'entità manager implementati. Questo è reso necessario soprattutto per problemi dovuti alla completa percezione dell'ambiente e agli angoli ciechi che i sensori installati su un singolo veicolo non riescono a percepire. Questo tipo di entità introdotta come manager non per forza di cose obbliga ad un modello di coordinazione completamente centralizzato ma sicuramente li abilita ed instrada verso tale soluzione. Si possono comunque mantenere schemi di comunicazione decentralizzata senza nessuna entità con il potere decisionale di accettare o declinare il passaggio di un veicolo dando compiti per lo più passivi alle infrastrutture intelligenti introdotte per aiutare il compito di cooperazione dei veicoli autonomi.

2.2.2 Schemi di comunicazione per sistemi intelligenti

Sono state definite le entità che compongono il MAS, sono con loro stati definiti anche i vincoli che devono rispettare per poter varare nuovi modelli di accesso alle intersezioni ed è stato infine specificato come tali modelli siano appartenenti alle dimensione dell'interazione, cioè a quella dimensione caratteristica e propria dei sistemi intelligenti distribuiti che dona loro la proprietà di non composizionalità.
La non composizionalità si riferisce alla caratteristica dei sistemi composti da più parti di riuscire ad avere un insieme di proprietà che non sono descrivibili nei termini delle parti stesse. Queste proprietà emergono dalle interazioni a basso livello fino a definire un comportamento globale dell'intero sistema. La comunicazione all'interno di un sistema complesso deve essere per definizione non banale, una sintassi ed una semantica vanno definite così come regole che instaurano l'apertura di un dialogo tra due agenti. Senza considerare i meccanismi che permettono di creare un canale di co-

municazione tra due agenti reali, esistono a grandi linee due grandi famiglie di policy di coordinazione [21]:

- Centralizzati: Questo tipo di policy implica che un agente venga eletto come master all'interno di una porzione del sistema e si riserva la capacità decisionale in base allo stato globale del sotto-sistema di cui ha controllo. Tale agente diventa di conseguenza lo stesso intersection manager il quale implementa una specifica policy e comunica ai veicoli in avvicinamento come e quando l'intersezione sarà assegnata a loro e potranno attraversarla. In questo caso esiste una gerarchia tra agenti che vengono divisi in ruoli differenti e viene parzialmente limitata l'autonomia dei veicoli.

- Decentralizzati: Gli agenti coordinano la propria attività in maniera indipendente e distribuita, comportandosi tutti come pear per raggiungere l'obiettivo comune. Le decisioni di controllo sono distribuite tra i vari agenti, senza nessun leader designato e di conseguenza la policy è locale ad ogni agente che agisce conservando totalmente la propria autonomia. La conoscenza è divisa tra i vari agenti che per coordinarsi necessitano di una fitta rete di interazioni.

Non esiste a priori un approccio che possa essere definito migliore dell'altro, ognuno porta con sé difetti sia nella ricerca della soluzione migliore sia nella maniera in cui le performance strettamente computazionali possono degradare. Un approccio centralizzato permette ad un'intersezione intelligente di mettere insieme, attraverso ogni singola informazione ricevuta dai veicoli in accesso, un quadro generale del problema così da poter arrivare a trovare la soluzione migliore possibile secondo la policy con cui è stata programmata. Un solo nodo decisionale ha bisogno di un numero di risorse elevato per poter operare sollevando il problema del singolo punto di rottura. D'altra parte in una soluzione decentralizzata si deve usare la sola interazione per riuscire ad ottenere gli elementi necessari a soddisfare i requisiti per la policy di coordinazione di ogni agente. Questa fitta rete introduce un overhead dovuto ai protocolli di comunicazione ma si dimostra molto robusta ed efficacie lasciando la libertà di scelta ad ogni singolo agente. Questa libertà di scelta non va comunque confusa con possibilità di poter agire al di fuori dei canoni di sicurezza che la mobilità veicolare necessita.
La decentralizzazione della coordinazione apre la possibilità ai progettisti di

realizzare policy complicate a piacere e con proprietà adattive ad ogni singola situazione di traffico e congestione ma sempre e comunque con meccanismi che garantiscano l'adesione ai requisiti definiti per i modelli di coordinazione per l'accesso alle intersezioni.

2.2.3 Scenari e sviluppi futuri

Le policy di coordinazione per veicoli autonomi sfruttano le potenzialità offerte dalle Vehicular-ad-hoc-Networks, grazie a ciò il volume di dati richiesto, generato e collezionato crescerà in maniera esponenziale. L'aumento del volume di dati ha reso necessario come mostrato nel primo capitolo l'evoluzione delle VANET nella più moderna concezione di Internet of Vehicle, la cui architettura fornisce i mezzi per il trattamento di informazioni e dati che meglio catturano i requisiti posti dalla sfida tecnologica posta. Da questo passaggio nasce la considerazione che i dati prodotti, elaborati e consumati da tutti gli attori in gioco all'interno di queste reti possano aderire ai requisiti delle "5V" [16] giustificando uno sviluppo futuro in cui molti servizi legati ai servizi di trasporto intelligente saranno forniti da un'elaborazione figlia di informazioni trattate con tecniche Big Data. Nell'elaborato in particolare vengono trattate le micro-interazioni tra veicoli, a granularità più fine le comunicazioni producono molte informazioni dal significato valido solo a livello locale. Se queste informazioni vengono incanalate attraverso i layer dell'IoV allora le tecniche di estrazione di conoscenza dai dati potranno portare benefici anche a livello globale, benefici che potranno essere tradotti in nuove opportunità non solo per le tecniche di routing dei veicoli all'interno dei percorsi stradali o bilanciamento del traffico attraverso una distribuzione dei flussi più cospicui ma anche e soprattutto occasioni per il business ed i servizi offerti a coloro che saranno trasportati dai mezzi intelligenti a guida autonoma. A tal proposito si riporta l'esempio di un'applicazione di servizi industriali per l'elaborazione dei dati prodotti dall'Internet of Vehicles denominata *CarStream* [23]. Questa applicazione è un primo esempio, testato su auto con conducente, di ciò che potrà accadere in futuro anche nello scenario contraddistinto dal dislocamento prima progressivo e poi totale di veicoli a guida autonoma. CarStream e tutte le applicazioni di questo tipo non saranno promotrici di sviluppo per quanto riguarda il miglioramento delle tecniche di comunicazione V2V bensì avranno un ruolo fondamentale per l'integrazione dei flussi di dati all'interno delle tecnologie cloud-based.

In particolare l'applicazione in oggetto è stata protagonista di test su larga scala connettendo un insieme di 30.000 veicoli su 60 diverse città, questi veicoli appartengono ad una flotta di mezzi offerti attraverso servizi di car sharing presenti in Cina con modalità non troppo differenti da quelle, per esempio, di Uber. Ognuno dei veicoli trasmette flussi di dati in maniera continua che comprendono informazioni raccolte attraverso la campionatura di un insieme elevato di sensori diversi che indicano lo stato del veicolo e le attività del guidatore. I flussi sono raccolti da server back-end che, dopo aver elevato il livello di qualità dei dati fino a farli diventare vere e proprie informazioni, offriranno servizi ai veicoli stessi seguendo il modello "Everything as a Service" (XaaS). L'abilità dei data analyst è in questo caso un valore aggiunto che permette di costruire grandi strutture di data warehousing che, in accordo con tutte le norme a protezione della privacy dei cittadini, permettono la vendita di servizi a compagnie commerciali per usi più svariati. Si possono pensare sistemi di promozione personalizzati attraverso gli strumenti di infotainment del veicolo: uno scenario plausibile è quello in cui le informazioni estratte attraverso le tecniche appena citate di Big Data o sistemi OLAP rivelino una certa esigenza periodica di un guidatore di fare fermate alle coordinate alle quali è presente un negozio che vende articoli di un certo tipo. E proprio attraverso i sistemi multimediali del veicolo, una società che paga per l'accesso ai dati potrà mostrare esattamente articoli del tipo che plausibilmente interessano il soggetto. In alternativa altre funzioni possono riguardare esigenze di sicurezza e di aiuto nei confronti delle forze dell'ordine, con la possibilità di individuare, sempre a causa della privacy, non chi commette infrazioni ma quali tratti stradali sono soggetti a guida pericolosa.

Capitolo 3

Simulazione per la coordinazione di veicoli autonomi

Il caso di studio considerato, come analizzato nel capitolo precedente, si configura in un sistema complesso multi-agente costituito da una società di veicoli autonomi immersi nell'ambiente e regolati da policy di coordinazione che a loro volta regolano la dimensione delle interazioni.
Un sistema complesso è condizionato nella sua evoluzione temporale sia dalle condizioni iniziali sia dalle perturbazioni che si manifestano durante l'esecuzione, il numero di agenti è elevato ed ognuno incide attraverso le proprie intenzioni sullo sviluppo futuro. Il numero di diverse combinazioni tra agenti ed interazioni è esponenziale e gli stati raggiungibili non sono verificabili a priori rendendo la comprensione e la verifica dell'esecuzione un'attività complessa.
Le caratteristiche che accomunano i sistemi complessi vanno da un comportamento non lineare, ad un'abilità nell'adattarsi ed evolversi alle mutate condizioni di esecuzione fino alla robustezza e all'autonomia a livello globale. Questo insieme di proprietà permettono lo studio del loro comportamento, della loro struttura e delle interazioni inter-agente quasi solo esclusivamente attraverso la **simulazione**.
La simulazione è un approccio allo studio del comportamento di un sistema che si contrappone al metodo scientifico tradizionale ove mancano le condizioni ed i presupposti per poterlo applicare. Non è possibile nel nostro caso

osservare e riprodurre una situazione sperimentale per i seguenti motivi:

- Non esistono ad oggi le possibilità fisiche di riprodurre in maniera sperimentale lo scenario previsto.
- Esistono necessità di scalare il sistema nel tempo e nelle dimensioni.
- Semplici schemi di interazione tra entità differenti possono portare a comportamenti globalmente diversi e complessi, rendendo impossibile a priori predire l'evoluzione del sistema.

Queste problematiche devono essere risolte adottando un approccio differente per lo studio dell'evoluzione di un sistema garantendo le seguenti funzionalità e proprietà:

- permettere di predire il comportamento;
- esplorare domande che non trovano risposta da metodi sperimentali;
- applicare l'analisi *"what if"* agendo sui parametri di esecuzione senza la necessità di porre condizioni sperimentali complesse o pericolose.

Requisito fondamentale ed indispensabile per poter fare l'analisi di un sistema attraverso la simulazione è un modello, cioè qualsiasi cosa attraverso il quale l'esecuzione di un esperimento per un certo sistema possa essere applicato al modello ottenendo risultati validi per la controparte reale che viene modellata. Ovviamente non è quasi mai possibile catturare tutti gli aspetti del sistema che viene modellato, per questo motivo si parla di processi di astrazione e rappresentazione nella progettazione di un modello. Le tecniche applicate, che vanno dall'aggregazione, alla semplificazione, all'omissione, devono catturare tutta la dinamica del processo che si sta modellando.
Stiamo parlando in questo momento del campo della simulazione in silicio, sempre più utilizzata in ambiti diversi che sfruttano la velocità e l'economicità di tale approccio soprattutto in fase di raccolta, elaborazione e confronto dei dati. Il tipo di simulazione varia in base alla granularità degli elementi considerati e per il caso del traffico veicolare autonomo, la necessità che si manifesta è quella di disporre di un ambiente in cui sia possibile replicare da un parte il comportamento di ogni singolo veicolo e

dall'altra ogni interazione tra di essi. Parliamo in questo caso di simulazione a micro-livello. L'obiettivo è duplice: capire attraverso la simulazione a micro-livello come varia il deflusso delle correnti di traffico rispetto ai metodi tradizionali per veicoli non connessi e dall'altra parte studiare se le micro-interazioni comportano un'emergenza di pattern di comportamento al macro-livello. Il capitolo si svilupperà introducendo per ogni sua sezione una delle tecnologie scelte per comporre lo stack tecnologico attraverso il quale è stata implementata la simulazione del sistema multi-agente di veicoli a guida autonoma.

3.1 SUMO

Simulation of Urban MObility (SUMO) è uno dei più importanti software di simulazione per il traffico veicolare, la sua prima versione è rilasciata nel 2002 dal Centro Aerospaziale Tedesco (DLR) [5].
La caratteristica principale di questo simulatore di traffico è quella di essere totalmente open source. Negli anni i continui sviluppi hanno portato alla creazione ed al rilascio di un insieme di applicativi che ora costituiscono un'intera suite la quale va oltre la simulazione e supporta l'integrazione sia con diversi linguaggi di programmazione sia con diversi tool per la raccolta dei dati e la loro elaborazione.
SUMO ha la caratteristica di essere molto versatile e, pur venendo concepito per la simulazione di reti stradali di grandi dimensioni, si adatta a qualsiasi contesto. La modellazione non si limita alle sole autovetture bensì vengono contemplati scenari di gestione del trasporto pubblico (anche ferroviario), i motocicli ed i pedoni stessi, riducendo la granularità dell'ambiente fino al singolo essere umano. Il modulo di interesse per lo scopo dell'elaborato si limita comunque alla simulazione microscopica dei flussi di traffico che sono resi possibili grazie alla modellazione di ogni veicolo in maniera individuale all'interno della rete ed ognuno mantiene una propria posizione ed una velocità istantanea ad ogni step di simulazione. I valori di queste misure sono calcolati da SUMO ad ogni passo, la durata del passo di avanzamento della simulazione può essere scelto e regolato in base alle esigenze del task in esecuzione.
La gestione del tempo è di conseguenza fatta a *tempo discreto* in uno spazio continuo. La dinamica dei veicoli simulati da questa applicazione è del tutto

simile a quella di veicoli condotti da essere umani, e non pilotati da software autonomi, quindi ci sono limitazioni alle prestazioni massime ottenibili da certi approcci di coordinazione ed accesso alle intersezioni dovute alla cosiddetta velocità di sicurezza che distanzia due veicoli per uno spazio che considera i tempi di reazione umani alla frenata. Se questa è una limitazione, esistono invece altri vantaggi che hanno fatto propendere la scelta verso questa opzione tra le altre possibili (esistono infatti molti altri simulatori altrettanto validi):

- I veicoli vengono introdotti all'interno della simulazione con l'obiettivo di raggiungere il traguardo del loro spostamento attraverso un percorso calcolato a priori, così come farà in futuro un veicolo a guida autonoma. Esiste la possibilità di un rerouting dinamico che dipende dallo stato di congestione delle strade che compongo l'itinerario del veicolo. Questa opzione non è qui considerata, e si presta bene per lo studio di algoritmi di bilanciamento dei flussi e lo studio del comportamento al macro-livello.

- Durante la simulazione i veicoli aderiscono a vincoli e limitazioni della rete stradale, proprietà fondamentale e necessaria per potersi focalizzare sulla sola coordinazione.

L'utilizzo è disponibile in due diverse modalità: una versione a linea di comando che si occupa di far progredire l'evoluzione del sistema ed una versione che utilizza un'interfaccia grafica con un rendering passo passo della disposizione sulle rete stradale di ogni veicolo. La versione con interfaccia grafica realizzata attraverso librerie openGL 3.1 mette a disposizione anche una serie di opzioni per poter interagire con la simulazione durante il suo svolgimento. Le prestazioni del simulatore offrono la capacità di gestire fino a 10.000 strade contemporaneamente, con una velocità di aggiornamento fino a 100.000 veicoli al secondo e una compatibilità con altri applicativi per il supporto della comunicazione V2X (Veins, iCS, VSimRTI).

3.1.1 Creazione rete stradale e veicolare

SUMO per iniziare una simulazione necessita di diversi file di configurazione, tutti in formato XML. Il file principale è nel formato *.sumocfg* e di seguito ne è riportato un esempio 3.1 .

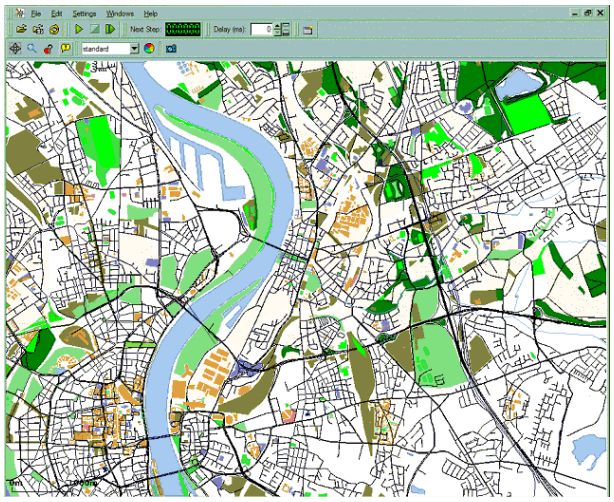

Figura 3.1: Esempio SUMO GUI.

Listing 3.1: *.sumocfg

```
<configuration>

 <input>
 <net-file value="cross.net.xml"/>
 <route-files value="cross.rou.xml"/>
 <additional-files value="cross.add.xml"/>
 </input>
 <report>
 <verbose value="true"/>
 <no-step-log value="true"/>
 </report>
 <tripinfo-output value="output.xml"/>

</configuration>
```

Come si intuisce dalla consultazione della struttura del file precedente, è caratterizzato da una parte di input che si compone essenzialmente di altri tre file in formato XML:

- *net-file*: è l'input che descrive la parte statica della rete. La rete stradale è un grafo orientato in cui ogni nodo (specificato nel file.*nod.xml*) è un'intersezione, cioè un incrocio tra strade. Ad ogni strada viene

associato un arco del grafo orientato (definito dal file *.edg.xml*). Ogni tipo di incrocio può ulteriormente essere modellato attraverso le specifiche del file in formato *.con.xml* che rappresenta le possibili manovre all'interno dell'intersezione stessa. Ogni intersezione ed ogni arco possono essere di tipo diverso, esistono varie politiche per governare le precedenze e diversi tipi di strade per quanto riguarda corsie, limiti di velocità e tipologie di veicoli ammessi ad ognuna di loro. Tutte queste opzioni possono essere configurate attraverso il file in formato *.typ.xml*. La complessità cresce notevolmente al crescere della rete rappresentata, di conseguenza nel tempo sono stati sviluppati tanti tool che rendono più agevole la loro compilazione e soprattutto la conversione nel file che riassume tutta la rete. Il tool più utilizzato, ed utilizzato anche per questo elaborato, è sicuramente NETCONVERT. Esso ha a disposizione tante altre opzioni non utilizzate e sicuramente ottime per simulazioni reali perchè danno la possibilità di convertire porzioni di rete stradale reale direttamente da *OpenStreetMap* [OSM] fino ad un file di rete SUMO. L'input da fornire al tool NETCONVERT, che è un'applicazione a linea di comando, è un file di configurazione nel formato *.netc.cfg* contenente tutti i riferimenti ai descrittori della rete e dei nodi.

- *route-files*: questo input si riferisce alla parte denominata **domanda di mobilità**. Essa altro non è che l'insieme di file descrittori delle entità che popoleranno la simulazione. Attraverso la parametrizzazione degli attributi di questi input si determinano i flussi di traffico sia veicolare che non, e quindi la frequenza dei pedoni e delle tipologie di utenti della strada compresi, ad esempio, i motocicli. La personalizzazione è possibile anche dei veicoli stessi, si può agire su tutti i parametri che determinano l'accelerazione, la decelerazione, gli errori di guida (o nel nostro caso gli errori di valutazione del contesto da parte del software autonomo che per ipotesi ha il controllo del veicolo) e la distanza minima tra due veicoli. Se, come nel caso in analisi, non hanno importanza i singoli veicoli ma i flussi di veicoli allora le opzioni di default semplificano il lavoro e la domanda di mobilità si riduce alla dichiarazione di flussi con un percorso prestabilito nella meta ma non nell'itinerario come mostrato di seguito.

Listing 3.2: *.rou.xml

```
<routes>
    <vehicles>
        <flow id="XX" begin="tStart" end="tEnd" from="
            startEdge" to="endEdge" probability="flowRate"
            />
    </vehicles>
</routes>
```

- *additional-files*: questo tipo di input specificato attraverso file di estensione *.add.xml* non sono necessari per la simulazione bensì regolano vari aspetti di quest'ultima, in particolare agendo sui meccanismi di raccolta dati e di regolazione dei deflussi alle intersezioni, se tali meccanismi sono basati su dispositivi semaforici. Di seguito è mostrato l'additional file utilizzato per garantire ad un'intersezione fasi semaforiche alternate.

Listing 3.3: *.add.xml

```
<additional>
  <tlLogic id="0" type="static" programID="my_program"
      offset="0">
    <phase duration="33" state="GGGrrrGGGrrrrrr"/>
    <phase duration="6" state="yyyrrryyyrrrrrr"/>
    <phase duration="33" state="rrrGGGrrrrrrrrr"/>
    <phase duration="6" state="rrryyyrrrrrrrrr"/>
    <phase duration="33" state="rrrrrrrrGGGGGG"/>
    <phase duration="6" state="rrrrrrrrryyyyyy"/>
 </tlLogic>

</additional>
```

Una volte completate le fasi di definizione dell'input, avendo quindi a disposizione sia una rete su cui effettuare la simulazione sia la domanda di mobilità, è possibile procedere con la simulazione vera e propria. Non si tratta di ciò che avviene nel nostro caso perché infatti questo compiuto rimane solo il primo tassello all'interno dello stack tecnologico che andremo a definire.

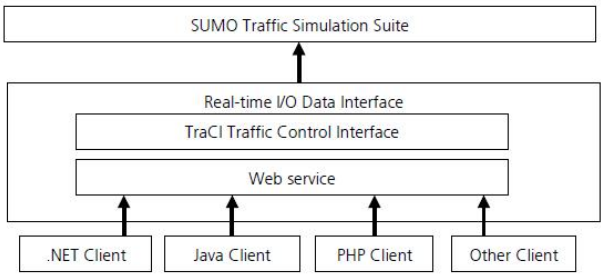

Figura 3.2: Architettura web service SUMO - TraCI.

3.2 TraaS

Il tool di simulazione microscopica SUMO offre la possibilità di accedere alla simulazione di traffico a runtime attraverso l'interfaccia I/O denominata TraCI (Traffic Control Interface). Il funzionamento segue un'architettura di tipo client/server in cui SUMO riveste il ruolo del server a cui diversi applicativi possono accedere per interagire con la simulazione attraverso l'interfaccia TraCI con il ruolo di client. L'opzione che avvia SUMO in modalità server è *–remote-port <INT>* (INT sta per il numero di porta TCP su cui SUMO si pone in ascolto di richieste client) e comporta il solo caricamento in memoria della rete e delle opzioni per la simulazione dei veicoli. È il client che collegandosi attraverso la porta TCP controllerà l'esecuzione della simulazione influenzando il comportamento dei singoli veicoli avendo anche accesso a tutti i dati che riguardano l'ambiente simulato. Come già detto il tempo viene modellato in maniera discreta, di conseguenza l'avanzamento temporale viene gestito attraverso un offset di valore fisso che rappresenta il passo di simulazione e viene impartito da un comando client. TraCI è un'interfaccia di comunicazione che può essere utilizzata con un client Python incluso nella suite di simulazione, ma per rendere l'interazione con SUMO più confortevole e user-friendly sono stati sviluppati dei Web Service attraverso il protocollo *SOAP (simple object access protocol)* a cui sono agganciati a loro volta dei client sviluppati scegliendo il linguaggio di programmazione in maniera libera e più adatta al tipo di stack tecnologico necessario per il task di simulazione in oggetto fig.[3.2].

Il tipo di client scelto per l'interazione con il simulatore nel lavoro in

oggetto è stato quello Java per ragioni motivate per lo più dall'utilizzo della piattaforma JASON per la creazione di un ambienti MAS e la programmazione di agenti autonomi. Una prima implementazione Java per interagire con SUMO fu il progetto *traci4j*, il quale si dimostrò molto affidabile soprattutto per il meccanismo integrato dello stesso linguaggio di programmazione per la gestione delle eccezioni. Traci4j utilizzava solo alcune delle opzioni messe a disposizione dall'interfaccia TraCI ma si distingueva tra i vari client per le buone prestazioni. Così ben presto un'estensione di nome *TraCI as a Service* (TraaS) [1] venne implementata offrendo un range completo di funzioni che incapsulano tutta la comunicazione attraverso l'interfaccia TraCI in maniera trasparente al programmatore. La connessione e la simulazione in generale vengono incapsulate in un'istanza della classe SumoTraciConnection la quale attraverso la chiamata a due semplici metodi, `do_job_get(SumoCommand cmd)` e `do_job_set(SumoCommand cmd)`, si occupa della richiesta di servizi in real-time all'istanza di simulazione. Da questo meccanismo deriva il nome TraCI as a Service. Di seguito è riportato un esempio di utilizzo della libreria in oggetto, in particolare la funzione per poter aggiungere un nuovo veicolo alla simulazione.

Listing 3.4: Aggiunta veicolo in simulazione SUMO attraverso libreria TraaS

```
1  public class Main {
2
3      static String sumo_bin = "c:/Program Files (x86)/sumo/
           bin/sumo-gui.exe";
4      static final String config_file = "simulation/config.
           sumo.cfg";
5
6      public static void main(String[] args) {
7
8          //start Simulation
9          SumoTraciConnection conn = new SumoTraciConnection(
               sumo_bin, config_file);
10
11         //set some options
12         conn.addOption("step-length", "0.1"); //timestep 100
               ms
13
14         try{
15
16             //start TraCI
17             conn.runServer();
```

```
18
19              //load routes and initialize the simulation
20              conn.do_timestep();
21
22              for(int i=0; i<3600; i++){
23
24                                      //current simulation time
25                                      int simtime = (int) conn.
                                            do_job_get(Simulation.
                                            getCurrentTime());
26
27                  conn.do_job_set(Vehicle.add("veh"+i, "car",
                        "s1", simtime, 0, 13.8, (byte) 1));
28                  conn.do_timestep();
29              }
30
31              //stop TraCI
32              conn.close();
33
34          }catch(Exception ex){ex.printStackTrace();}
35
36      }
37
38  }
```

3.3 Agenti BDI

Un aspetto cruciale per la simulazione di modelli di coordinazione per veicoli a guida autonoma è la modellazione stessa dei veicoli su ambiente computazionale con l'obiettivo di ricreare il sistema multi-agente più possibilmente fedele a quella che si immagina sarà la realtà delle cose in un futuro prossimo. Un agente software è un'entità computazionale in grado di fornire le funzionalità per cui viene concepito in maniera autonoma ed in un ambiente ricco di interazioni ed input con le seguenti quattro proprietà:

- autonomia;
- proattività;
- reattività;
- socialità.

Esistono diverse architetture per agenti software ma quella scelta, motivata dall'integrazione con il linguaggio Java attraverso il framework JASON è l'architettura BDI (Belief-Desire-Intention) che nasce sulla spinta dei sistemi intenzionali, così definiti dal filosofo Daniel Dennet [6]

> "...whose behavior can be predicted by the method of attributing belief, desires and rational acumen."

Il modello seguito per concepire quest'architettura è quello del comportamento umano guidato nelle sue azioni esattamente dalle tre attitudini che si vogliono rappresentare come stati mentali per agenti autonomi:

- Belief: informazioni che l'agente ha del mondo.
- Desire: sono gli stati del mondo a cui l'agente aspira ad arrivare.
- Intention: desideri o azioni che l'agente ha preso l'impegno di soddisfare

La differenza che esiste tra i secondi ed i terzi sta nel dettaglio che la collezione dei desideri di un agente rappresenta tutti possibili elementi che ne influenzano il comportamento, bensì nessun desiderio è nulla di più di un'opzione fin quando non viene eletto ad intenzione. Ogni intenzione esprime un obiettivo che determina le azioni dell'agente nei confronti degli altri agenti e dell'ambiente. Il processo decisionale che elegge un goal da perseguire e le azioni da compiere per ottenere l'obiettivo è denominato *practical reasoning* ed in particolare si compone di due fasi distinte che sono la fase di *deliberation* e di *means-end reasoning* (fig. 3.3).

La prima delle due si occupa di eleggere le intenzioni cioè di scegliere quali debbano essere i goal che si vogliono ottenere mentre la seconda va alla ricerca del modo in cui raggiungerli. Il modo che un agente ha per perseguire un obiettivo eletto ad intenzione è la selezione di uno più *plan* dalla *plan library* che rappresenta l'insieme della conoscenza pratica dell'agente stesso. Il funzionamento del meccanismo di reasoning è regolato dal ciclo di controllo mostrato in fig. 3.4 in una sua versione semplificata, ρ indica il risultato dell'attività di sensing che ha l'obiettivo di aggiornare i belief. L'attività di reasoning procede con la fase di deliberation che dà come risultato l'intenzione eletta ad obiettivo da perseguire attraverso l'esecuzione dell'azione Π [3].

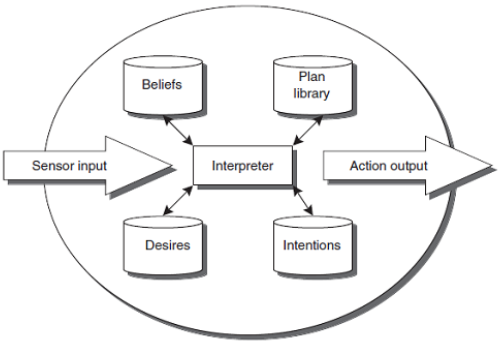

Figura 3.3: Procedural reasoning system (PRS).

Un altro aspetto cruciale che va al di fuori della sola modellazione interna della struttura di un sistema decisionale è la dimensione dell'interazione e la comunicazione tra agenti, basata tipicamente sulla *speech-act theory* [20] che si basa sul presupposto che gli atti di comunicazione non servano solo per esprimere o descrivere un contenuto bensì siano paragonabili a delle vere e proprie azioni in grado di esercitare un particolare influsso sul mondo circostante.

3.3.1 Jason

Jason è una piattaforma per lo sviluppo di sistemi multi-agente, in particolare un'estensione del linguaggio orientato agli agenti chiamato *AgentSpeak*. Jason è sviluppato in Java ed utilizzato per programmare il comportamento individuale degli agenti. È al pari di SUMO un software open source distribuito sotto licenza GNU LPGL con plugin disponibili per gli IDE più utilizzati nella programmazione Java come Eclipse o jEdit.
AgentSpeak è ispirato alla programmazione logica, e le informazioni sono rappresentate in forma di *predicati* che possono specificare semplici proprietà (ad es. predicato raining) oppure delle relazioni che esistono e che si verificano tra uno o più oggetti (ad es. *likes(John, pizza)* esprime i gusti di John). Senza fare una vera e propria introduzione alla programmazione logica, Jason dà la possibilità di specificare il comportamento di un agente, in termini di beliefs, goals e plans, attraverso la definizione di un file **.asl* caratterizzato da una sintassi Prolog-like.

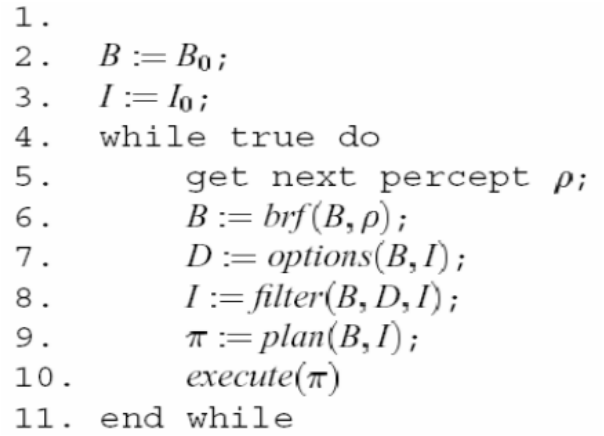

Figura 3.4: Control loop agente BDI.

Beliefs e Goals sono esprimibili nel file con estensione .asl, e per quanto riguarda i goal esiste una distinzione che permette di indicare preceduti da punto esclamativo gli *Achievement Goals* (!g(t_1 ,..,t_n)) e da punto interrogativo i *Test Goals* (?g(t_1 ,..,t_n)).
Il reasoning cycle di Jason è una specializzazione di quello mostrato in fig. 3.4 e una modifica nella base di conoscenza o nello stato di un goal generano un evento che necessita attraverso una procedura di reazione di essere servito. Si parla di questi eventi come di *triggering events* e sono riassumibili in ognuno dei casi in cui venga aggiunto o cancellato un belief od un goal. Il meccanismo con cui si reagisce ad un evento e si descrivono le azioni necessarie per servirlo sono i *plans* che nel linguaggio AgentSpeak puro hanno la seguente sintassi:

triggering_event: context ← body.

- Il **triggering_event** denota il tipo di evento che lo specifico piano deve gestire.

- Il **context** rappresenta le circostanze necessarie, in termini di congiunzioni di letterali, per cui uno specifico piano possa essere attivato per lo specifico triggering_event.

- Il **body** rappresenta la sequenza della azioni che devono essere eseguite nel momento in cui il contesto viene verificato e il piano attivato dall'agente.

L'insieme dei plans rappresenta il *know-how* dell'agente, le "ricette" che l'entità autonoma ha a disposizione per raggiungere i propri goals.

La potenza di Jason risiede nei meccanismi con cui va ad estendere AgentSpeak, ed in particolare nell'utilizzare Java a basso livello per implementare azioni interne che vanno ad influenzare l'ambiente.
Le estensioni principali sono le seguenti:

- Annotazioni: sono la nozione attraverso cui i predicati vengono arricchiti di ulteriori informazioni. Un'annotazione comunemente utilizzata, e che presentano di default tutti i predicati nella forma [source(S_i)], mette al corrente l'agente su chi abbia generato l'informazione che il predicato stesso sta esprimendo. Questo semplice meccanismo amplia notevolmente le possibilità di definizione di contesti differenti per gestire gli eventi differenziando il comportamento in base alla fonte dell'informazione.

- In Jason i beliefs e le rispettive annotazioni possono essere pre-processate con regole in stile Prolog.

- Azioni interne (*Internal Actions*) rappresentano in Jason il meccanismo molto potente che incapsula le interazioni con gli oggetti Java. Queste interazioni sono cruciali per l'invocazione di sistemi legacy, per l'utilizzo degli artefatti in un sistema di tipo *Agent & Artifact* oltre che per l'effettiva realizzazione di azioni pragmatiche da parte degli agenti.

Le azioni degli agenti sono di due tipi:

- Azioni pragmatiche: sono le azioni che vanno ad inficiare sull'ambiente cambiandone lo stato. In Jason sono realizzibili attraverso le internal action (userLibrary.userAction(X,Y,R) con X e Y parametri e R che unifica con il valore di ritorno dell'azione) che gestiscono la chiamata al metodo execute delle classi Java che a loro volta estendono la classe *DefaultInternalAction*.

- Azioni di comunicazione: sono le azioni che permettono la creazione di socialità tra gli agenti e si realizzano utilizzando le primitive di comunicazione messe a disposizione da Jason stesso. L'invio del messaggio è garantito dall'azione *.send(B,ilf,m(x))* utilizzata per l'invio ad un singolo agente specificato nel parametro B, o dall'azione *.broadcast(ilf,m(x))* per l'invio all'intero sistema.
  La ricezione è gestita esattamente come un triggering event, e per identificare il mittente vengono utilizzate le annotazioni; la forma è la seguente:

$$+\mathrm{m(x)[source(A)]} : \mathrm{context} \leftarrow \mathrm{body}.$$

Per poter costruire e mettere in funzione un MAS è necessario specificare qualche sorta d'ambiente dove situare gli agenti. Qui Jason offre la possibilità di crearne uno simulato che perfettamente si incastra con le esigenze che scaturiscono dai requisiti dell'elaborato. Il tutto è fatto utilizzando sempre Java, un ambiente Jason è realizzato estendendo la classe *Environment* come mostrato nell'esempio 3.5

Listing 3.5: Esempio di Environment

```
1  import jason.*;
2  import ...;
3  public class myEnv extends Environment{
4      ....
5      public myEnv() {
6          Literal loc = Literal.parseLiteral("location(3,5)");
7          addPercept(pos1);
8      }
9      public boolean executeAction(String ag, Term action) {
10         if (action.equals(...)) {
11             addPercept(ag,
12             Literal.parseLiteral("location(souffle,c(3,4))");
13         }
14         ...
15         return true;
16     }
17 }
```

Come si vede è possibile in maniera abbastanza agevole aggiungere percezioni ambientali attraverso il metodo *addPercept(String Agent, Literal Percept)* e, trovandosi all'interno di ambiente Java, sfruttare tutte le potenzialità offerte dal linguaggio di programmazione.

3.4 Integrazione dello stack tecnologico

Le parti presentate nelle sezioni precedenti sono quelle che andranno a comporre l'architettura costruita per la simulazione di traffico autonomo. La loro composizione produce l'architettura finale così come mostrato in fig. 3.5:

- **.asl*: Il comportamento intenzionale di ogni tipo di agente autonomo coinvolto nel progetto di una simulazione può essere implementato attraverso la definizione di un file in Agent Speak Language. L'infrastruttura Jason che si occupa della gestione del ciclo di vita di ogni agente rende agevole l'inserimento di un numero arbitrario di veicoli replicando la stessa struttura intenzionale per ognuno di loro e gestendone il ciclo di reasoning. Viene creato un flusso di controllo indipendente per ogni entità autonoma all'interno della stessa simulazione.

- *Java Environment*: Come mostrato nel frammento di codice 3.5, Jason promuove l'integrazione con il linguaggio Java e fornisce classi predefinite che, se estese, facilitano le interazioni tra agente ed ambiente.
  Non avendo a disposizione un ambiente reale su cui poter sperimentare i risultati degli algoritmi di coordinazione che si vorranno testare, l'implementazione della logica dell'ambiente simulato risulta fondamentale. Questo è il layer centrale dell'architettura e di conseguenza qui va implementata la logica degli algoritmi di coordinazione.
  Le strutture dati che vengono predisposte a questo livello devono raccogliere le informazioni dal tool di simulazione e creare le percezioni ambientali fornendole agli agenti che simulano il comportamento dei veicoli autonomi. Il compito dell'environment non si ferma comunque a questo: per i già citati algoritmi di coordinazione centralizzata, è necessario definirne la logica sotto forma di intelligent intersection. L'entità intelligente, reificata su una simulazione di questo tipo, diventa un insieme di costrutti Java che incapsulano al loro interno l'algoritmo oggetto della simulazione. Questo meccanismo permette, modificando la sola logica dell'algoritmo stesso, di mantenere quasi del tutto inalterata la struttura degli agenti intenzionali. Il discorso è diverso invece per gli algoritmi di tipo distribuito e decentralizzato in cui

la capacità decisionale viene ascritta agli agenti stessi. La simulazione di tali algoritmi prevede la definizione di azioni interne richiamate dagli agenti stessi nel modo che viene specificato direttamente nel file *.asl.

- *SUMO*: L'utilizzo di un tool open-source di simulazione come SUMO offre una serie di vantaggi non indifferenti. I più tangibili sono quelli che riguardano la gestione del livello di presentazione, della parte grafica, della dinamica e della fisica dei veicoli. L'insieme di tutta l'offerta tecnologica, insieme all'opzione di interazione in stile client-server, marca una linea netta tra quelli che sono i compiti del programmatore e quelli che non lo sono. Le intenzioni elette dagli agenti Jason che governano il comportamento di ogni veicolo sono infatti elaborate dal layer centrale che le commuta in comandi che rispettano il formato richiesto dal server SUMO. Ogni volta che ogni veicolo avrà elaborato la propria intenzione allora uno step di simulazione potrà essere eseguito. Le norme stradali, che vanno dal rispetto della distanza di sicurezza alla velocità massima da tenere su ogni strada, sono prerogative di un veicolo a guida autonoma che nel caso specifico vengono delegate al simulatore stesso. L'insieme di queste caratteristiche si sposa con quelli che sono gli obiettivi dell'elaborato: concentrarsi esclusivamente sulla logica di coordinazione.

La comunicazione tra i layer dell'architettura è di tipo bidirezionale, come sottolineato dalle connessioni presenti nei blocchi della fig. 3.5. Questa caratteristica deriva dall'analisi del problema eseguita in prima istanza: ogni agente autonomo situato nel proprio ambiente ne percepisce la dinamica e può agire su di esso. La successiva scelta di adozione dell'infrastruttura Jason per la gestione del Multi-Agent-System ha indotto l'adozione di un'architettura di progetto a tre livelli che colma *l'abstraction gap* che si crea nel tentativo di modellare l'ambiente attraverso sistemi di tipo intenzionale. Di seguito è analizzata la dimensione delle interazioni:

- **.asl* $\leftrightarrow$ *Java environment*: Le interazioni tra agenti ed ambiente sono bidirezionali ma non simmetriche per il tipo di architettura di progetto scelta. L'astrazione di ambiente catturata dal layer centrale infatti viene by-passata dal meccanismo delle *Internal Action* che sfruttano le features della libreria TraaS. L'istanza della simulazione viene

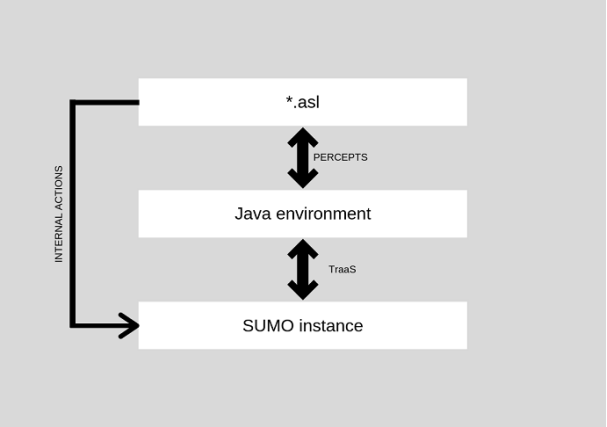

Figura 3.5: Architettura progetto.

incapsulata all'interno di un oggetto della classe *SumoTraciConnection* utilizzata attraverso il pattern singleton in maniera Thread-Safe. Ogni agente potrà agire sull'ambiente in maniera diretta attraverso richieste di servizi all'oggetto stesso. La collezione di Internal Action in questo modo simula il bagaglio di azioni che un veicolo reale compie sull'hardware.
L'elaborazione guidata dalla logica dell'algoritmo di coordinazione per l'attraversamento all'intersezione è gestita, come già detto, dal layer centrale. Esso ha il compito di tradurre l'output in dei percept che saranno dati in input ai veicoli i quali saranno responsabili, attraverso il loro ciclo di ragionamento, di eleggere l'intenzione che si concretizzerà nell'adozione di un certo plan.

- *Java environment* $\leftrightarrow$ *SUMO instance*: Un'istanza SUMO in esecuzione è un bacino di informazioni enorme a cui l'astrazione che simula l'ambiente deve attingere per poter applicare le politiche di coordinazione. La fisica dei veicoli, la dinamica dei movimenti, i dati relativi alle loro posizioni insieme ai flussi informativi provenienti dai sensori che è possibile dislocare lungo la rete stradale, sono elaborati e resi disponibili dall'istanza SUMO. Essendo il simulatore basato su una gestione del tempo di tipo discreto, il meccanismo è piuttosto logico: per prima cosa va scelto un intervallo di durata di uno step che non

sia né troppo corto né troppo lungo. Nel primo caso si avrebbe un inutile sovraccarico sulle risorse computazionali mentre nel secondo si correrebbe il rischio di eseguire una campionatura sui dati che farebbe perdere importanti eventi. Una volta fissato questo parametro il layer responsabile dell'astrazione dell'ambiente di simulazione estrae le informazioni di interesse, e lo fa mandando una serie di richieste al server SUMO che risponde con messaggi che aderiscono al protocollo SOAP.

- *Comunicazione inter-agente*: la comunicazione inter-agente in questo scenario di simulazione corrisponde all'interazione Vehicle-to-Vehicle in un caso di studio reale. Non viene considerato il tipo di protocollo utilizzato per una futura realizzazione di tale feature all'interno dell'Internet of Vehicles bensì vengono sfruttate le internal action presenti di default all'interno dell'infrastruttura Jason. Nella creazione di un'architettura per la simulazione di veicoli a guida autonoma questo aspetto risulta fondamentale per l'implementazione di politiche di coordinazione decentralizzata. Un limite non superabile risulta comunque quello della completa eliminazione di un'intermediario di comunicazione che promuova lo scambio degli identificativi dei veicoli. Nessuno tra i tool utilizzati simula o abilita la simulazione di protocolli che contemplino per esempio l'individuazione automatica di veicoli nelle vicinanze, una delle funzionalità indispensabili per l'implementazione di protocolli senza manager.

Per ridurre i tempi di implementazione di un nuovo algoritmo di coordinazione sono state predisposte funzioni di libreria che, analizzando i file di configurazione della simulazione, estraggono le strutture dati legate alle intersezioni e ai sensori ad esse legati. Molti dei servizi offerti dalla libreria TraaS sono di basso livello: questa caratteristica amplia le possibilità di personalizzazione e di controllo della simulazione dei veicoli ma ne complica la programmazione. Nell'integrazione dei layer dell'architettura si è deciso quindi di fornire funzioni di più alto livello che combinano i risultati di più chiamate di servizi in un'unica chiamata di procedura. Un esempio è la funzione che permette di ricavare la direzione di svolta come mostrato nel frammento di codice 3.6.

Listing 3.6: Funzione per ricavare la direzione di svolta di un veicolo

```
1  public static Direction getTurningDirection(String vehicle,
       String lane) throws Exception{
2          SumoLinkList links = (SumoLinkList) simulation.
               do_job_get(Lane.getLinks(lane));
3          String route = simulation.do_job_get(Vehicle.
               getRouteID(vehicle)).toString();
4          SumoStringList edges = (SumoStringList) simulation.
               do_job_get(Route.getEdges(route));
5          String currentEdge = simulation.do_job_get(Vehicle.
               getRoadID(vehicle)).toString();
6          String tmpEdge = "";
7          for(int i = 0; i < edges.size()-1; i++){
8              if(edges.get(i).equals(currentEdge)){
9                  tmpEdge = edges.get(i+1);
10             }
11         }
12         /*I can get the turning direction by looking for the
               direction of the
13          * link that connects the incoming and the outgoing
                arch
14          */
15         final String nextEdge = tmpEdge.split("_")[0];
16         Optional<SumoLink> direction = links.stream().filter(
17                 el -> el.notInternalLane.split("_")[0].equals(
                       nextEdge)).findFirst();
18         if(direction.isPresent()){
19             if(direction.get().direction.equals("r"))
20                 return Direction.RIGHT;
21             else if(direction.get().direction.equals("l"))
22                 return Direction.LEFT;
23             else if(direction.get().direction.equals("s"))
24                 return Direction.STRAIGHT;
25         }
26         return Direction.NULL;
27     }
```

Infine un'ultima considerazione va fatta sulle interazioni tra intersezioni intelligenti e veicoli: la logica che molti algoritmi seguono prevede che siano i veicoli in avvicinamento ad informare, se presente, l'entità di coordinazione. Essa, dopo un'elaborazione basata sulla policy che regola l'attraversamento, fornisce indicazioni al veicolo su che approccio adottare per eseguire l'operazione in sicurezza. Per come è strutturato il flusso di dati tra i vari

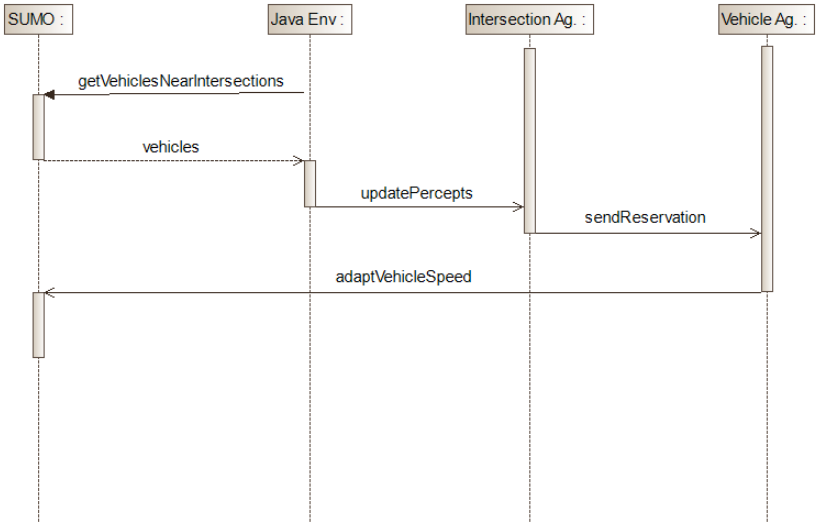

Figura 3.6: Esempio di interazione tra agenti, ambiente e SUMO.

layer dello stack (fig.3.6) non si riescono ad inserire veicoli all'interno della simulazione che in maniera proattiva eseguano questa operazione, e la causa è dovuta alla tipologia di sensori che si possono utilizzare, limitati ad una serie di dispositivi dislocati lungo la rete stradale e non direttamente installati sui veicoli stessi.

Capitolo 4

Simulazione e analisi dei risultati

Una volta definita l'architettura di progetto per la simulazione della coordinazione di veicoli autonomi, bisogna eleggere sia un algoritmo centralizzato sia uno distribuito per l'implementazione e l'analisi dei risultati effettuando un confronto con le tecniche di gestione della congestione stradale per veicoli a guida umana. In ognuna delle prove sono state fatte le seguenti assunzioni semplificative:

- Solo un flusso entrante per volta ha diritto ad accedere all'intersezione. Questo non cambia la complessità del problema, argomento trattato in [8, 11].
- I flussi entranti all'intersezione sono costituiti da un minimo di 2 ad un massimo di 4, ognuno di essi è limitato ad una sola corsia.
- Sono ammessi tutti i tipi di percorso dal momento che il diritto di accesso è ristretto ad un veicolo per volta. Non ci sono divieti di svolta o inversione.
- Tutte le corsie che compongono la rete stradale hanno un limite di velocità uguale e fissato a 60 km/h. I veicoli, essendo autonomi, seguono il limite posto e viaggiano di conseguenza a velocità uguale e costante quando non sono nei pressi di un'intersezione.

La simulazione verrà eseguita sulla rete rappresentata in fig. 4.2, e i veicoli saranno inseriti in maniera casuale al suo interno con con una probabilità

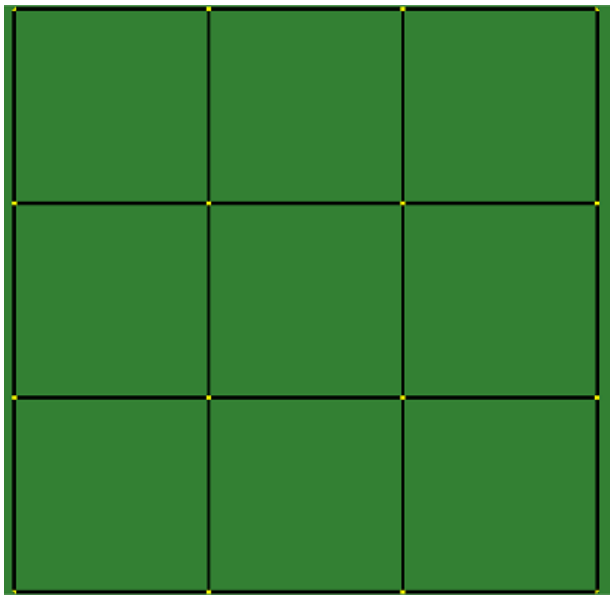

Figura 4.1: Rete stradale utilizzata per la simulazione.

arbitraria per testare le prestazioni di ognuno degli algoritmi di coordinazione. Ogni veicolo inserito ha una percorso predefinito e scelto sempre in maniera casuale tra un insieme di 20 predefiniti e di lunghezza diversa. Questo tipo di semplificazioni permettono di avere una fase di analisi ed implementazione più semplice ma non sminuiscono la portata del problema. Non è irrealistico lo scenario in cui un veicolo che cerca di completare un tragitto nel minor tempo possibile raggiunga in ogni circostanza i limiti previsti dalla strada che percorre.

4.1 Simulazione di veicoli con conducente

Prima di passare all'analisi degli algoritmi implementati per la simulazione di veicoli a guida autonoma, viene innanzitutto presentata un'analisi di sperimentazioni effettuate utilizzando lo stesso tipo di scenario ma popolato da veicoli simulati con un comportamento che ricalca quello dei veicoli con conducente. Vengono eliminate le possibilità di interazione tra i veicoli, e l'esecuzione della simulazione avviene senza la creazione di un agente Jason

per ogni entità. Eliminando queste caratteristiche, l'architettura presentata precedentemente collassa su due soli layer:

- Java Environment: La programmazione dell'ambiente Java in questo tipo di simulazione si limita alla sola gestione dell'ingresso di nuovi veicoli al suo interno (compresa l'assegnazione della loro rotta di viaggio) e alla gestione del tempo. Poche e semplici operazioni effettuate attraverso le API della libreria TraaS permettono la realizzazione del comportamento desiderato.

- SUMO: viene mantenuto il tool di simulazione già presentato per la gestione della dinamica dei veicoli, e in questo caso assume anche maggiore importanza perché in assenza di capacità decisionale propria dei veicoli i soli impianti semaforici saranno responsabili dell'accesso alle intersezioni.

L'assenza degli agenti in questo tipo di simulazione comporta un'interazione ridotta al solo utilizzo delle API necessarie per la comunicazione tra Java e SUMO (con uno schema di tipo Client/Server come già mostrato nel capitolo precedente), non sono più necessari i meccanismi che implementano la proprietà di consapevolezza del contesto propria degli agenti autonomi e nessun tipo di logica applicativa deve essere aggiunta a livello di gestione dell'accesso alle intersezioni. Ad ogni intersezione viene infatti installato un semaforo attraverso gli input che vengono forniti nel file in formato *.add.xml* a SUMO stesso, è qui che vengono decisi i cicli che contraddistinguono le fasi semaforiche: dato che per avere una possibilità di confronto delle prestazioni, è necessario attenersi alle assunzioni esemplificative poste all'inizio del capitolo, i semafori presenti hanno un ciclo contraddistinto da un numero di fasi pari al numero di flussi entranti in ogni intersezione. Durante ogni fase uno solo dei flussi di veicoli avrà diritto all'attraversamento dato che ogni flusso è composto esattamente da una corrente e questo comporta, in assenza di divieti di svolta (altra assunzione), che ogni corrente è in incompatibile. Queste simulazioni saranno eseguite variando la durata delle fasi semaforiche e della portata dei flussi di traffico prima di procedere ai test che chiameranno in causa i veicoli autonomi e connessi, i risultati quindi ottenuti sono un ottimo benchmark per la misura del miglioramento ottenuto dai test successivi.

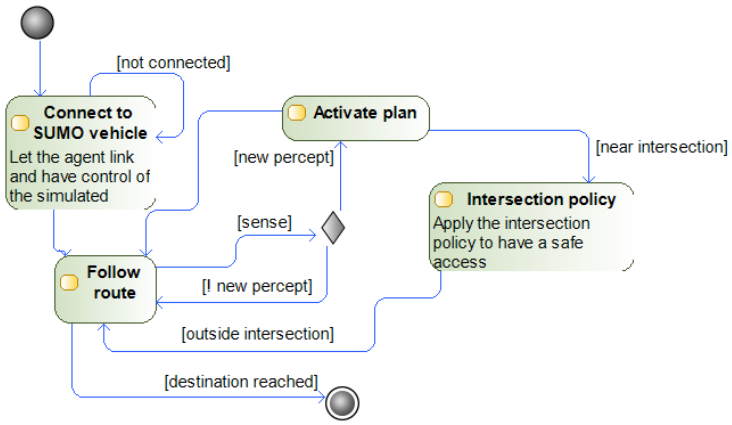

Figura 4.2: Comportamento veicolo simulato.

4.2 Simulazione di veicoli a guida autonoma

4.2.1 Reservation-based Intersection Control Mechanism

Il primo algoritmo per l'accesso alle intersezioni in un contesto di guida con veicoli autonomi eletto per essere implementato ed eseguito attraverso la struttura di progetto introdotta nel capitolo precedente, è quello ispirato al lavoro alla policy di coordinazione centralizzata che ha la propria peculiarità nelle prenotazioni [11].
L'assunzione di base fatta per rendere possibile l'implementazione di questa policy per l'accesso ad un'intersezione è che sia resa possibile la comunicazione wireless tra gli agenti in esecuzione, deve essere inoltre presente un'entità centrale (già descritta come Intelligent Intersection) la quale è responsabile della gestione dei permessi di attraversamento concessi ai veicoli entranti. Le auto saranno autorizzate ad entrare nell'area di contesa solo e soltanto nei momenti in cui ne avranno, attraverso il protocollo, avuto il permesso. La gestione dei permessi non fa decadere i requisiti di autonomia dei veicoli: anche la guida così come è consuetudine al giorno d'oggi ha una gestione simile attraverso le regole stabilite, per esempio, dai semafori ma la persona che governa il funzionamento del veicolo è comunque considerata autonoma e responsabile delle proprie azioni.

La simulazione che sperimenta la distribuzione di prenotazioni per l'accesso all'intersezione necessita di definire sia il comportamento dell'agente che governa il sistema nel suo intero sia quello degli agenti che hanno il controllo dei veicoli.

- **Comportamento del sistema**: questo tipo di policy di coordinazione viene gestita da un'entità centrale intelligente che deve essere introdotta per permettere ai veicoli di manifestarsi in richiesta dello spazio che necessitano. Il veicolo entrante contatta l'intersezione attraverso un messaggio che indica il tempo di arrivo, la velocità e la direzione insieme alle caratteristiche tecniche che specificano le prestazioni massime in termini di accelerazione e velocità raggiungibile. L'intersezione riesce, grazie alle informazioni ricevute e alla conoscenza completa dello scenario in cui la richiesta si inserisce, a riservare uno slot temporale in cui il veicolo potrà attraversa l'incrocio in sicurezza

- **Comportamento dei veicoli autonomi**: un veicolo prosegue il suo tragitto attraverso le sue caratteristiche di autonomia, che lo guidano in sicurezza percorrendo le strade che compongono il percorso. In prossimità degli incroci l'insieme dei sensori che donano consapevolezza del contesto ambientale non possono, senza interazioni, guidare con la stessa sicurezza il veicolo. In questo tipo di policy allora è lo stesso agente che, rilevando la distanza al di sotto di una certa soglia in avvicinamento all'incrocio, dà inizio allo scambio di messaggi manifestando la necessità di uno slot temporale. Fino a quando questa autorizzazione, accompagnato dallo slot richiesto, non viene concessa il veicolo continua l'avvicinamento come se dovesse eseguire un arresto completo. Quando infine una prenotazione è concessa dall'entità responsabile della gestione del sistema, allora il veicolo adotta una velocità di crociera tale per cui la percorrenza dello spazio restante all'intersezione necessiterà esattamente del tempo che intercorre tra il rilascio della prenotazione e il suo inizio.

L'effettiva implementazione dell'ambiente di simulazione per questo algoritmo di coordinazione dei flussi entranti in un'intersezione ha seguito il seguente processo:

1. Il primo passo sta nel completare l'analisi delle azioni di cui necessita ogni singolo veicolo per poter essere inserito all'interno della simulazione: per questo e per tutti gli altri algoritmi che si potrebbero

implementare ogni agente che controlla un veicolo all'interno della simulazione è dotato di operazioni che gli permettono di fare un arresto al prossimo incrocio, di ripartire in sicurezza e avvicinarsi a velocità costante fino ad essere in un punto predefinito a tempo stabilito.

2. Successivamente è stato definito il file *.asl* (fig. 4.1) per programmare il comportamento degli agenti che avranno controllo dei veicoli nella simulazione.

3. Come terzo passo infine deve essere definito l'environment Jason responsabile della gestione delle interazioni tra gli agenti e della corretta percezione del contesto ambientale per questi ultimi. All'interno di questa fase viene definita la logica di coordinazione e di conseguenza anche il comportamento di tutti gli incroci intelligenti. Sono i dati estratti tramite le API fornite da SUMO che rivelano le informazioni sui veicoli entranti alle intersezioni le quali propongono una reservation al veicolo che in maniera autonoma regola la propria velocità per rispettarla. Viene quindi calcolato il tempo di arrivo di ogni veicolo all'intersezione a cui si appresta ad accedere e viene concessa una reservation con i seguenti criteri:

   - Se l'intersezione non ha al momento rilasciato prenotazioni per l'attraversamento in sicurezza, allora il veicolo in avvicinamento concorda il passaggio al tempo di arrivo stimato mantenendo la sua velocità costante.
   - Se esiste contesa per l'attraversamento l'intersezione concorda, in ordine di corsia e di distanza, una prenotazione con ogni veicolo. Al veicolo n-simo sarà concessa una prenotazione la quale inizia con un offset temporale rispetto a quella precedente pari al tempo di attraversamento del veicolo precedente alla sua velocità stimata.

Listing 4.1: car.asl

```
1 !init.
2
3 /* Plans */
4
5 +!init: true <- .print(self, "has been deployed!").
```

```
6
7  +msg(NAME)
8      : true
9      <-    -msg(NAME);
10           +id(NAME);
11           !follow(route);. //messaggio di partenza
12
13
14 +!follow(route)
15     : not stop(INTERSECTION) & id(NAME)
16     <- .print("Following route, Vehicle: ", NAME).
17
18 +stop /* emergency stop signal */
19     : id(NAME)
20     <-     car.stop(NAME);
21         .send("master",tell,msg(done(NAME)));
22         -stop[source(master)].
23
24 +slowDown(TIME,ID,CTIME)
25     : id(NAME) &  not gotReservation & not routeTerminated
26     <-  car.slowDown(NAME,TIME,CTIME);
27         -slowDown(TIME,ID,CTIME);
28         +gotReservation.
29
30 +slowDown(TIME,ID,CTIME)[source(percept)]
31     : id(NAME) &  not gotReservation & routeTerminated
32     <-  -slowDown(TIME,ID,CTIME)[source(percept)];
33         +gotReservation.
34
35 +outSideJunction(ID)[source(percept)]
36     :id(NAME) & not routeTerminated
37     <- car.acelerate(NAME);
38         -outSideJunction(ID)[source(percept)];
39         -gotReservation.
40
41 +outSideJunction(ID)[source(percept)]
42     :id(NAME) & routeTerminated
43     <-  -outSideJunction(ID)[source(percept)];
44         -gotReservation.
45
46 +freeRoad(NAME) //end of emergency stop
47     : true
48     <- car.resume(NAME);
49         !follow(route).
```

4.2.2 Decentralized Autonomous Intersection Access Control (DAIAC)

Il sistema di controllo di accesso presentato in precedenza è solo uno dei tanti tentativi di definire delle policy regolate da un'entità di controllo centrale. In contrapposizione esistono altre soluzioni che evitano tutti i difetti legati agli alti costi di realizzazione delle infrastrutture intelligenti e fanno emergere vantaggi anche in termini di sicurezza e praticità della soluzione. Per questa famiglia di policy di coordinazione è stato deciso di implementare per eseguire simulazioni e prove l'algoritmo decentralizzato presentato alla 17° conferenza internazionale sui sistemi di trasporto intelligente (ITSC) [8] denominato DAIAC. La caratteristica fondamentale di questo schema è quella di permettere ai veicoli in avvicinamento all'intersezione di decidere sull'accesso basandosi completamente sulla percezione del contesto resa completa nel suo insieme dallo scambio di beacon in maniera passiva. Questo significa che vengono eliminate le interazioni favorendo la sicurezza e la protezione della privacy mantenendo un elevato livello di efficienza. È fondamentale, dato che il potere decisionale rimane decentralizzato, creare un modello condiviso tra i veicoli che ne detta le linee guida nel comportamento.
L'intersezione viene definita come mostrato in figura 4.3 e hanno un'importanza fondamentale le seguenti aeree, valide per ogni corsia in entrata:

- *Contention zone*: è l'area in avvicinamento all'intersezione che ha lo scopo di permettere ai veicoli di determinare se esiste una contesa con altri provenienti da corsie differenti.

- *Stop zone*: è l'area ulteriormente più vicina all'incrocio che allo stesso tempo permette la fermata in sicurezza del veicolo nel caso in cui l'elaborazione concluda che sia necessario farlo.

- *Safety zone*: è l'area identificata dall'algoritmo DAIAC che più si trova vicina all zona di intersezione dei flussi. All'interno di questa area non è possibile iniziare e completare una fermata in sicurezza, di conseguenza nell'esecuzione dell'algoritmo la necessità di concedere la precedenza ad altri veicoli in conflitto deve essere decisa con un margine maggiore.

Dato che i veicoli in avvicinamento non possono essere consapevoli, se non attraverso qualche forma di interazione, delle informazioni legate alla

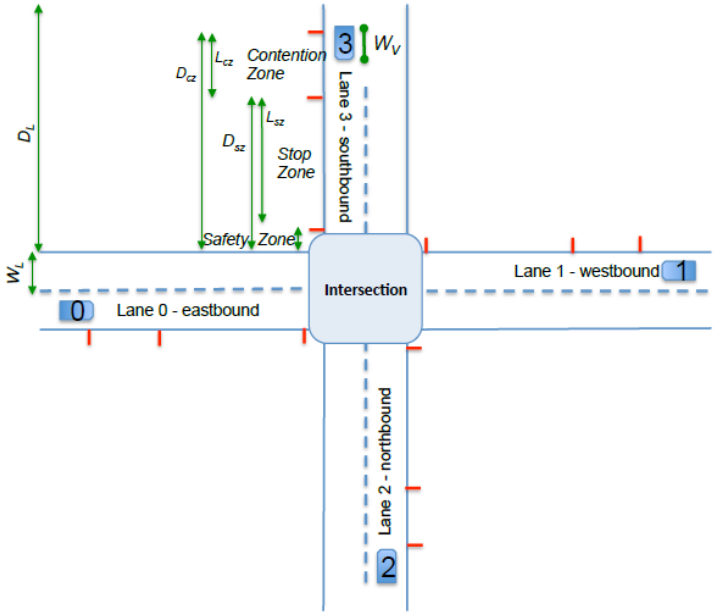

Figura 4.3: Modello dell'intersezione.

presenza di contese, allora un meccanismo di esposizione di tali informazioni deve essere reso disponibile. Per diminuire al minimo le interazioni, dei semplici sensori di presenza installati lungo le carreggiate permettono la costruzione dell'unità informativa denominata *beacon* in maniera molto semplice: ogni copia viene inviata a tutti i veicoli che si trovano in una delle tre zone appena elencate ad ogni time step. Il beacon ha un'utilità molto importante anche per quanto riguarda la sincronizzazione dei clock dei veicoli e contiene le seguenti informazioni:

- Un *timestamp* che indica il momento esatto del rilascio dell'informazione, è infatti di fondamentale importanza dare una "fotografia" del contesto indicando il tempo in cui è stata catturata considerati possibili ritardi nella ricezione.
- Una struttura dati che indica la presenza o l'assenza di veicoli in ognuna delle tre aree per ognuna delle corsie entranti.
- Un valore per la durata del *verde logico* ed uno che indica invece il valore della durata della fase di *all red* che sono determinanti, per stabilire in presenza di contesa, chi ha precedenza in un dato momento temporale.

L'algoritmo distribuito, come già accennato nel contenuto informativo del beacon, ha una funzione che simula un segnale attuato in maniera virtuale. Quindi ogni veicolo è in grado localmente di calcolare il valore di tale segnale attraverso la procedura mostrata nel listato 4.2. τ indica la frequenza a cui viene ricalcolato e inviato il beacon mentre la durata della fase di green e di all red permettono attraverso semplici calcoli di determinare quale flusso ha logicamente il segnale verde. È fondamentale la fase in cui nessuna corsia ha via libera all'ingresso nell'intersezione perché si attua una guardia a protezione di piccoli sfasamenti di clock che determinerebbero nei casi limite il passaggio contemporaneo di flussi incompatibili.

Listing 4.2: Funzione per il calcolo del segnale attuato

```
1 public static boolean getTrafficSignal(int time,String
      lane_id,int[] presence_s,int green,int allRed
2           , int tau, SumoTraciConnection sumo, String vehId)
               throws Exception{
3         Double direction = (Double) sumo.do_job_get(Vehicle.
              getAngle(vehId));
```

```
4         int angle = direction.intValue();
5         boolean new_period = false;
6         int period = green + allRed;
7         int epoch = time / period;
8         int prev_epoch = (time - tau) / period;
9         if(epoch > prev_epoch){
10           new_period = true;
11        }
12        int phase_id = epoch % 4;
13        if(new_period && presence_s[phase_id] == 0){
14           for(int i = 1; i < 4; i++){
15              int next_id = (phase_id + i) % 4;
16              if(presence_s[next_id] == 1){
17                 phase_id = next_id;
18                 break;
19              }
20           }
21        }
22        int index = 0;
23        switch(angle){
24           //SUMO angle matching DAIAC index
25           case 0:
26              index = 2;
27              break;
28           case 270:
29              index = 1;
30              break;
31           case 180:
32              index = 3;
33              break;
34           case 90:
35              index = 0;
36              break;
37        }
38        if(index == phase_id){
39           int t = time - epoch * period;
40           if(t <= green){
41              return true; //GREEN
42           }else{
43              return false;
44           }
45        }else{
46           return false;
47        }
48     }
```

Il calcolo della funzione precedente è solo il primo passo per la completa esecuzione del DAIAC, insieme a questo servono due ulteriori output ottenuti nel seguente modo:

- Calcolo della contesa: una semplice funzione deve calcolare, in base ai flag scambiati nel beacon, se esiste contesa nel tentativo di attraversare l'intersezione e fornire in output un semplice valore booleano che lo indichi.
- Valutazione della fermata: ogni veicolo quando esegue l'algoritmo DAIAC deve valutare se l'ipotesi di fermarsi al prossimo incrocio non sia impossibile per le caratteristiche del veicolo stesso. Viene quindi fatto un calcolo che valuta se sia possibile compiere un arresto in sicurezza e viene restituito un valore booleano che segnala ciò.

Premesso che il beacon viene inviato a tutti i veicoli che sono almeno vicini all'intersezione quanto è la lunghezza della contention zone, l'algoritmo per determinare se è necessario aspettare per attraversare l'incrocio viene eseguito dai veicoli nel momento in cui fanno il loro ingresso nella stop zone ed ha le caratteristiche mostrate nel listato 4.3. Vale anche in questo caso la considerazione sulla necessità dei veicoli di essere dotati di sensori per il calcolo delle distanze i quali permettano di inquadrarlo all'interno di una delle 3 aree di interesse. Essi sono simulati attraverso il layer dell'environment Jason il quale simula appunto tali sensori attraverso dei percept aggiunti agli agenti che governano il comportamento dei veicoli.

Listing 4.3: Algoritmo DAIAC

```
1  if(inStopZone.equals("true")){
2      if(!iscontending  || trafficSignal){
3          return true; // Procede Safely
4      }else {
5          return false;
6      }
7  }else{
8      if(!iscontending  || tooLate || trafficSignal){
9          return true; // Procede Safely
10     }else {
11         return false;
12     }
13 }
```

L'algoritmo DAIAC è implementato come azione interna per gli agenti definiti dal file nel listato 4.4 mentre, rispettando l'architettura di progetto già definita per la simulazione del traffico veicolare, il resto dei requisiti sono implementati nel layer centrale all'interno della definizione dell'environment Jason. L'unione di questi moduli permette l'esecuzione della simulazione attraverso SUMO di un contesto di guida in cui le entità sono veicoli autonomi coordinati con algoritmo passivo decentralizzato nell'accesso alle intersezioni.

Listing 4.4: carDAIAC.asl

```
1  !init.
2
3  /* Plans */
4
5  +!init: true <- .print(self, "has been deployed!").
6
7  +msg(NAME)
8      : true
9      <-    -msg(NAME);
10           +id(NAME);
11           !follow(route);.
12
13 +!follow(route)
14     : not stop(INTERSECTION) & id(NAME)
15     <- .print("Following route, Vehicle: ",NAME).
16
17 +stop //emergency stop signal
18     : id(NAME)
19     <-      car.stop(NAME);
20         .send("master",tell,msg(done(NAME)));
21         -stop[source(master)].
22
23
24 +freeRoad(NAME) //end of emergency stop
25     : true
26     <-   car.resume(NAME);
27          !follow(route).
28
29
30 +beacon(TIME, PC0, PS0, PC1, PS1,PC2, PS2, PC3, PS3,
       INCTNZONE, GREEN, ALL_RED)[source(percept)]
31     : distance(D) & id(NAME)
```

```
32        & not cardaiac.daiac(TIME, NAME, PC0, PS0, PC1, PS1,
              PC2, PS2, PC3, PS3,GREEN , ALL_RED ,false, D)
33        & not stopping
34        & not routeTerminated
35    <-  .print("Stopping, ", NAME, " ", D);
36        cardaiac.stop(NAME);
37        +stopping;
38        -resuming.
39
40
41 +beacon(TIME, PC0, PS0, PC1, PS1,PC2, PS2, PC3, PS3,
      INCTNZONE, GREEN, ALL_RED)[source(percept)]
42    : distance(D) & id(NAME)
43        & cardaiac.daiac(TIME, NAME, PC0, PS0, PC1, PS1,PC2,
              PS2, PC3, PS3,GREEN , ALL_RED ,true, D)
44        & resuming
45        & not routeTerminated
46    <-    true.
47
48
49 +beacon(TIME, PC0, PS0, PC1, PS1,PC2, PS2, PC3, PS3,
      INCTNZONE, GREEN, ALL_RED)[source(percept)]
50    : distance(D) & id(NAME)
51        & cardaiac.daiac(TIME, NAME, PC0, PS0, PC1, PS1,PC2,
              PS2, PC3, PS3,GREEN , ALL_RED ,true, D)
52        & not resuming
53        & stopping
54        & not routeTerminated
55    <- cardaiac.resume(NAME);
56        .print("Resuming, ", NAME, " ", D);
57        +resuming;
58        -stopping.
```

4.3 Valutazione dei risultati

Gli algoritmi descritti nelle due sezioni precedenti rappresentano due approcci di filosofia differente per affrontare il problema della coordinazione di veicoli autonomi alle intersezioni. L'esecuzione della simulazione attraverso l'architettura presentata deve essere analizzata in termini di prestazioni, e il metro di giudizio appropriato è il confronto con una simulazione che, nelle stesse condizioni, utilizza semafori o incroci a precedenza per decidere

$$\frac{\sum_{i \in V} (T_i - T_i^*)}{|V|}$$

Figura 4.4: Formula per il calcolo del ritardo medio.

l'ordine di accesso all'incrocio. Deve essere definita una misura per valutare i risultati e l'impatto che il traffico ha sul percorso di ogni veicolo. Quando si deve raggiungere una meta, l'obiettivo è quello di compiere il viaggio in sicurezza e nel minor tempo possibile. Il tempo passato nel traffico, ed in generale per gli spostamenti, vuole essere minimizzato. Per questo viene definita la metrica mostrata in fig. 4.4 grazie alla quale è possibile ricavare una misura che rappresenta le prestazioni medie dell'algoritmo di coordinazione riassunte nella misura del ritardo medio dei veicoli che partecipano alla simulazione. Il calcolo è abbastanza semplice, e si basa su una sommatoria del ritardo di ogni veicolo, calcolato attraverso una semplice differenza che confronta il tempo realmente impiegato con il tempo ottimale di viaggio. Il tempo ottimale di viaggio viene calcolato assumendo che il veicolo abbia una velocità di crociera costante e pari al limite di sicurezza imposto dalle strade che compongono il percorso esattamente in linea a quelle che erano le ipotesi fissate all'inizio del capitolo. Chiaramente i ritardi sul percorso si accumulano nell'attraversamento delle intersezioni, ed è qui che la loro gestione più sarà efficiente e più garantirà performance migliori.

I veicoli vengono creati ed inseriti all'interno della simulazione con un percorso prestabilito ed assegnato loro tra un pool di percorsi disponibili e di lunghezza variabile. SUMO, insieme alla sua installazione, mette a disposizione un insieme di frammenti di codice in linguaggio Python che rendono disponibile funzionalità diverse e molto comode in fase di creazione dell'ambiente di simulazione, una di queste è definita nel file *randomTrips.py* e crea un numero predefinito e dato in input di percorsi che attraversano la rete stradale fornita. Questo supporto ha creato i percorsi che in maniera casuale saranno poi assegnati ad ogni veicolo.
Dato che, come è stato già detto, la gestione del tempo all'interno del simulatore scelto è di tipo discreto allora va fissato un intervallo di tempo che

Figura 4.5: Simulazione dell'algoritmo DAIAC.

rappresenta lo step di simulazione. Il contesto del traffico evolve molto in fretta e gli eventi vanno campionati senza la possibilità di poterne perdere alcuno, quindi questo intervallo è di valore uguale a 50ms (cioè 20 campionature al secondo). Durante ogni timestep viene generato un numero casuale, questo numero viene poi confrontato con un valore di soglia arbitrariamente scelto: se il numero estratto è minore alla soglia allora un nuovo veicolo viene creato ed inserito all'interno della simulazione allo step successivo. Grazie a questo meccanismo, agendo sulla soglia, si creano dei flussi più o meno intensi di traffico e si possono studiare le risposte date dagli algoritmi di coordinazione messi alla prova. In particolare, utilizzando la classe *Random* fornita dalla librerie Java, si può facilmente generare un numero reale compreso tra 0 e 1 quindi ponendo la soglia pari a 0.1 si otterrà una possibilità di generazione pari al 10% per step. Variando il valore si varia la probabilità di pari passo. Nello specifico le prove di simulazione sono state eseguite con valori di probabilità di creazione di un veicolo avente i valori nell'intervallo [0.1;0.7], crescente a partire dall'estremo inferiore con un incremento di 0.1.

Le policy di coordinazione testate ed i valori fissati per i parametri sono invece i seguenti:

- AWS: questa simulazione si riferisce all'assenza di coordinazione semaforica in un contesto di guida con veicoli con conducente. Ogni intersezione è regolata dalla consueta precedenza a destra, simulata da SUMO.

- DAIAC: questa simulazione è quella che replica su ogni intersezione

AVG DELAY (s)	**Densità di traffico**						
Algoritmo	0.01	0.02	0.03	0.04	0.05	0.06	0.07
Reservation	**0.78**	**0.81**	**1.44**	14.84	31.33	**56.79**	74.34
DAIAC 5s	2.5	6.15	11.39	**11.52**	**26.73**	83.20	112.77
DAIAC 10s	2.83	8.77	15.04	17.63	44.87	63.25	79.63
DAIAC 15s	3.69	11.60	18.92	33.93	42.02	74.54	100.66
AWS	10.68	12.60	16.76	43.00	48.23	62.93	**74.31**
Semaphores 10s	79.68	88.08	169	196	204.51	328.16	368.78
Semaphores 20s	114.41	115.20	177.89	180.50	189.89	273.26	279.35

Tabella 4.1: Ritardi medi accumulati dai veicoli.

l'algoritmo introdotto nella sezione precedente. I parametri su cui si può lavorare per valutare l'efficienza e trovare il setting giusto sono la durata delle fasi di *green* e *all_ red*. L'ampiezza delle zone di contesa e di sicurezza sono state calcolate in accordo allo spazio di frenata necessario per completare l'arresto di un veicolo alla velocità massima consentita su quel particolare tratto stradale. Le fasi di verde sono fissate a tre valori differenti (5s, 10s, 15s), per capire in base al volume di traffico quale durata potrà essere migliore ed in generale come varia l'efficienza al variare dei parametri in gioco.

- Reservation: qui si simula invece l'efficienza nella coordinazione delle *Intelligent Intersections*, con agenti che hanno potere decisionale sull'accesso alle intersezioni e gestiscono l'assegnazione delle prenotazioni.

- Semafori: l'esecuzione di una simulazione il cui accesso è regolato dai comuni semafori è importante per fissare un benchmark riguardo le prestazioni degli altri approcci implementati. Le fasi per completare un ciclo sono state regolate a due diversi valori di durata: 10s e 20s.

Ogni simulazione studia sotto una certa configurazione di ogni policy l'andamento del traffico in un periodo temporale pari a 30 minuti e i risultati sui ritardi accumulati sono raccolti nella tabella 4.1.

Come si può immediatamente notare, non esiste mai un confronto tra i risultati ottenuti con gli algoritmi per veicoli connessi e quelli a segnalazione semaforica. Questo tipo di risultato era facilmente preventivabile, soprattutto per i livelli di traffico non elevato: i semafori offrono un tipo di

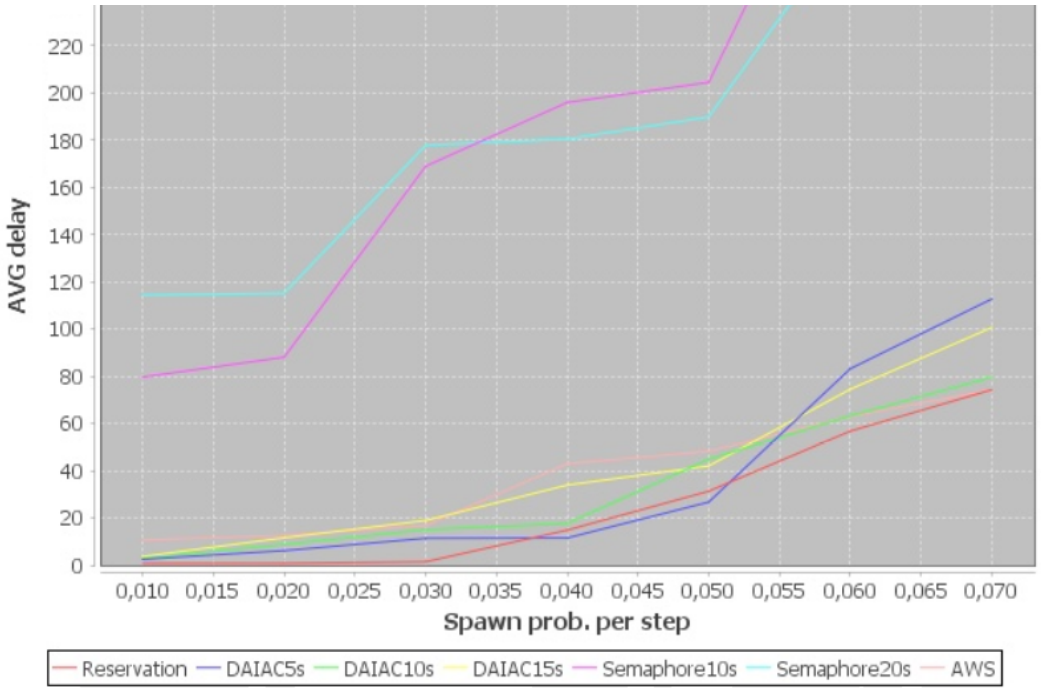

Figura 4.6: Grafico ritardi medi veicoli nelle simulazioni.

coordinazione statica che garantisce il rispetto dei canoni di sicurezza impedendo a flussi non compatibili di accedere simultaneamente all'intersezione ma in assenza di dati (raccolti e distribuiti in rete) la probabilità di essere fermati senza la necessità di doverlo fare aumenta in maniera drammatica per ogni veicolo che si approccia all'intersezione. La quasi assenza di ritardo per i due approcci studiati per veicoli connessi alle stesse condizioni è collegato alla possibilità di rendere consapevoli i veicoli in avvicinamento dell'assenza di contesa per il passaggio.

Il grafico in fig.4.6, mostra in maniera più evidente un altro aspetto messo in luce dai risultati ottenuti in condizioni di traffico crescente: la reazione sulle prestazioni quando la caoticità del traffico aumenta è ben diversa nei due algoritmi sperimentali rispetto a quella ottenuta dai semafori. Nel primo caso si può notare una sorta di linearità, o comunque una crescita costante del ritardo procurato ad ogni veicolo all'aumentare della percentuale di spawning dei veicoli durante la simulazione, mentre per quanto riguarda i

semafori abbiamo un accumulo di ritardo che supera il tempo ideale di percorrenza del percorso già a probabilità del 40% di creazione di un veicolo per step di simulazione. Infatti gli archi della mappa creata sono di lunghezza pari a 400 metri, ed ogni tragitto è composta da un numero di archi non inferiore a 4, richiedendo un tempo di percorrenza di almeno 145 secondi. Nessuno degli algoritmi per veicoli autonomi implementati si avvicina a quel valore, dunque si può affermare che le possibilità di coordinazione offerte dall'utilizzo di modelli di coordinazione avanzati per l'accesso e l'attraversamento degli incroci stradali permetterà di abbattere la soglia del 100% di ritardo medio accumulato rispetto al tempo di percorrenza ideale in ambito di traffico urbano. Due ulteriori considerazioni finali vanno fatte:

- Le prestazioni dimostrate da ogni singola simulazione contemplano tutte e tre le principali casistiche che lo studio del traffico deve affrontare. L'inizio vede un popolamento delle strade più o meno rapido, in accordo con la probabilità di creazione dei veicoli, che porta alla saturazione delle intersezioni. Quando il numero dei veicoli raggiunge il limite (fissato a 500) la creazione si ferma così da poter rendere l'analisi dei risultati comprensiva anche della fase di completo deflusso. Questo limite viene raggiunto, rientrando all'interno dei 30 minuti di simulazione, solo per valori di probabilità almeno al di sopra del 40

- Per quanto riguarda la temporizzazione delle fasi, in quei tipi di policy che prevedono una divisione temporale dei flussi in contesa, sempre dal grafico in fig. 4.6 si può apprezzare che fasi di durata contenuta tendono ad adattarsi bene in situazioni di traffico contenuto. Esse infatti si saturano molto prima causando ritardi dovuti alle manovre di frenata e ripartenza, discorso inverso se la scelta ricade su fasi troppo lunghe che penalizzano le situazioni con flussi entranti non eccessivi. Il vantaggio del DAIAC rispetto ai semafori risiede nella semplicità con la quale queste fasi possono essere modificate, cioè semplicemente cambiando un valore nel beacon inviato ad ogni veicolo che si trova all'interno della contention zone.

In un'ultima istanza, un'analisi va effettuata sui risultati ottenuti dai due algoritmi rivolti alla coordinazione di veicoli autonomi: la loro natura è differente per la differente attribuzione di potere decisionale alle entità in gioco. E i risultati riflettono questa caratteristica eleggendo quasi sempre

come approccio migliore quello centralizzato, regolato in questo caso da una policy basata sulle reservation. I vincoli di sicurezza e di condivisione dell'intersezione sono decisi a livello centrale ed i veicoli, essendo dotati di caratteristiche di autonomia, rispettano in maniera costante e minuziosa tutte le tempistiche previste. La coordinazione decentralizzata prevede l'evoluzione della simulazione guidata dalle decisioni singolarmente prese da ogni agente autonomo, il che aumenta il volume dell'interazione. All'aumentare delle interazioni il throughput dell'intersezione gestita da quella particolare policy di coordinazione diminuisce mentre altre proprietà caratteristiche della decentralizzazione del controllo emergono. In questo contesto, molto importante è la seguente considerazione: l'aumento dell'efficienza della coordinazione dei flussi di veicoli all'interno del traffico urbano ed extraurbano potrà essere un obiettivo ancor prima della completa realizzazione di un contesto in cui la penetrazione nel mercato dell'automobile di veicoli autonomi sarà realtà. E in un contesto di parziale penetrazione, sarà la coordinazione decentralizzata l'unica opzione possibile per far coesistere veicoli autonomi e non, mentre la rigorosità delle policy centralizzate non permetteranno un rispetto così stringente dei vincoli da parte dei conducenti, rendendo insicure le intersezioni stradali. Nel confrontare le prestazioni pure dei due algoritmi si può notare come in situazioni di traffico molto blando, il ritardo accumulato da quello centralizzato sia prossimo allo zero, mentre tende sempre più a conformarsi ai valori di ritardo che comporta l'algoritmo DAIAC quando le intersezioni vanno verso la saturazione. Questo mette in luce una grande flessibilità mostrata dall'agente centrale a cui è attribuito il compito di distribuire le reservation e, allo stesso tempo, una sorta di limite inferiore raggiungibile dall'accumulo dei ritardi dei veicoli dovuto alle ipotesi poste all'inizio del capitolo. Un valore di accumulo di ritardo a cui tutte le simulazioni con la stessa probabilità di generazione di veicoli tendono.

Capitolo 5

Conclusioni

L'elaborato ha analizzato quelle che sono le tecnologie la cui evoluzione porterà all'effettiva creazione dei primi mezzi di trasporto in grado di immergersi nell'ambiente urbano e in maniera autonoma condurre i passeggeri alle destinazioni desiderate. Dato che lo spostamento delle merci e delle persone è un problema che affligge la società per i motivi già citati nell'introduzione di questo lavoro, ogni possibilità offerta dalla tecnologia per attenuare la misura dei danni provocati a ogni singolo individuo è da indagare, studiare e arrivare alla conclusione sull'effettivo apporto migliorativo che porterà alla causa. L'analisi di una simulazione che si occupa della coordinazione di sistemi di trasporto intelligente già fa intravedere risultati che porteranno un miglioramento nell'efficienza della gestione del traffico. Capire se l'ordine di grandezza del miglioramento sarà effettivamente questo oppure no, e dare un'approssimazione di precisione alla simulazione posta in essere, non è un compito semplice:

- Le assunzioni decise come semplificazioni servono per mostrare il funzionamento degli algoritmi implementati, ma senza una controprova non si può affermare con certezza di quanto la realtà venga effettivamente semplificata.
- L'assenza di pedoni, ciclisti ed ogni altro utente della strada al di fuori degli autoveicoli, è un ulteriore motivo che rende inconfrontabili con certezza i risultati ottenuti.
- Nei casi reali, la sicurezza è un pilastro che detta linee guida nella progettazione sia dei mezzi autonomi, sia dei meccanismi che permettono

loro di coesistere all'interno del traffico. La sicurezza sotto questo punto di vista limita le possibilità di sfruttare a pieno tutte le opportunità che nascono dalla connessione dei veicoli in rete a favore di un margine maggiore del necessario e sufficiente così come le normative solitamente richiedono.

Oltre all'analisi dell'impatto in termini di prestazioni, andrebbe compiuta un'analisi che non rientra all'interno degli obiettivi dell'elaborato ma che risulta fondamentale per qualsiasi nuovo investimento: i costi. L'impatto economico del ricambio della flotta di veicoli è un'invariante all'interno della società così come è sempre presente a livello di investimenti pubblici una parte di fondi che servono per il rinnovo delle infrastrutture. Il passaggio non potrà essere drastico, ma ci sono buone possibilità che un periodo transitorio in cui coesisteranno veicoli con e senza conducente farà da apripista ad un nuovo ed innovativo modo di intendere la circolazione stradale. Come si può notare dall'analisi dei risultati delle simulazioni, i differenti meccanismi di coordinazione per l'accesso alle intersezioni dedicati a veicoli a guida autonoma, ottengono prestazioni migliori rispetto ai classici semafori per ogni situazione di traffico. La crescita dei ritardi medi risulta quasi lineare con l'aumento dei flussi sia per l'algoritmo con Intersection Manager che per quello decentralizzato. Questo porta a concludere che, a prescindere dall'algoritmo utilizzato, la penetrazione nel mercato di veicoli in grado di sfruttare a pieno la dimensione delle interazioni di un sistema multi-agente abilita la progettazione di modalità completamente innovative per la gestione del traffico e la prevenzione dei conflitti, le quali superano in maniera significativa i metodi classici ed utilizzati finora. L'efficienza è importante, ma anche altri aspetti vanno considerati:

- Oltre alla sicurezza informatica, dei software a cui viene delegato il potere decisionale con aspetti di completa autonomia, sarà sicuramente tema di dibattito sociale l'assegnazione di compiti dove un errore può essere fatale a più individui ad un solo agente senza assistenza umana.

- I test per l'omologazione dei moduli che piloteranno le automobili all'interno di contesti ambientali estremamente dinamici e complessi avranno caratteristiche non semplici da individuare e parametri che dovranno contemplare la valutazione della bontà delle scelte in scenari critici.

- La standardizzazione dei protocolli di coordinazione e cooperazione tra veicoli dovrà garantire un'uniformità di comportamento e meccanismi per la consapevolezza dei veicoli i quali dovranno essere aggiornati sempre alle ultime versioni presenti.

- I dati che vengono collezionati dall'utilizzo che ognuno di noi fa della tecnologia sono nella maggior parte delle volte elaborati a fini soprattutto commerciali per essere a loro volta prodotti molto ambiti dal mercato. Il tema che tiene banco in questi casi è la difesa della privacy dei cittadini, e la tracciatura di un confine che delimita l'identificazione dei dati del singolo consumatore o della tendenza della totalità dei consumatori.

Le simulazioni condotte attraverso l'implementazione dell'architettura progettata sono risultate in linea con i risultati teorici mostrati nelle letture svolte durante la scelta delle policy di coordinazione da testare, questo testimonia che la dinamica dei veicoli e la dimensione delle interazioni è stata catturata in maniera fedele dal modello creato.

Ringraziamenti

Ringraziando quelli che sanno di esserci sempre stati.

Bibliografia

[1] Traas documentation.

[2] Wu Lenan Abdeldime M.S. Abdelgader. The physical layer of the ieee 802.11p wave communication standard: The specifications and challenges. In *Proceedings of the World Congress on Engineering and Computer Science 2014 Vol II*, San Francisco, USA, 2014.

[3] Wooldridge Bordini, Hubner. *Programming Multi-Agent Systems in AgentSpeak using Jason*. 2007.

[4] Ing. Umberto Crisalli. Teoria del deflusso alle intersezioni, 2016.

[5] Laura Bieker-Walz Michael Behrisch Daniel Krajzewicz, Jakob Erdmann. Sumo, simulation of urban mobility: An overview. 2011.

[6] Daniel Dennet. *The Intentional Stance, p. 17*. 1987.

[7] Chris Valasek Dr. Charlie Miller. Remote exploitation of an unaltered passenger vehicle. 2015.

[8] Joud Khoury John Khoury. Passive, decentralized, and fully autonomous intersection access control. In *IEEE 17th International Conference on Intelligent Transportation Systems (ITSC)*. Victoria Transport Policy Institute, 2014.

[9] Sherali Zeadally Juan Contreras-Castillo and Juan Guerrero-Ibanez. Internet of vehicles: Architecture, protocols, and security. IEEE, 2016.

[10] Tasha Keeney. Mobility-as-a-service: why self-driving cars could change everything. 2017.

[11] Peter Stone Kurt Dresner. Multiagent traffic management: A reservation-based intersection control mechanism. Austin, TX 78712 USA, 2004. University of Texas.

[12] Todd Litman. Autonomous vehicle implementation predictions. Victoria Transport Policy Institute, 2018.

[13] Shaoshan Liu. Creating autonomous vehicle systems. 2016.

[14] Giovanni Pau Mario Gerla, Eun-Kyu Lee. Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*, Seoul, South Korea, 2014.

[15] Aditya Medury Offer Grembek, Alex Kurzhanskiy. An intelligent intersection. 2018.

[16] Vinita Jindal Punam Bedi. Use of big data technology in vehicular ad-hoc networks. In *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, New Delhi, India, 2014.

[17] Federico Casanello Carola Jorquera Rodrigo Fernandez, Eduardo Valenzuela. Evolution of the transyt model in a developing country. 2016.

[18] Jason Rowley. The well-funded startups driven to own the autonomous vehicle stack, 2018.

[19] SAE. Society of automotive engineers, 2009.

[20] John Searle. *Speech acts*. 1969.

[21] Swati Basak Vivek Kumar Singh, Neelam Modanwal. Mas coordination strategies and their application in disaster management domain. In *2011 2nd International Conference on Intelligent Agent and Multi-Agent Systems*. IEEE, 2011.

[22] Wikipedia. Mobile ad-hoc network.

[23] Xie Lin Liu Zhang, Wo. Carstream: An industrial system of big data processing for internet-of-vehicles. China, 2017.