Simulating Cellular Communications in Vehicular Networks:

making SIMULte *interoperable with* veins

Vehicles in Network Simulation

Giovanni Nardini, **Antonio Virdis**, Giovanni Stea

University of Pisa - Italy

Outline

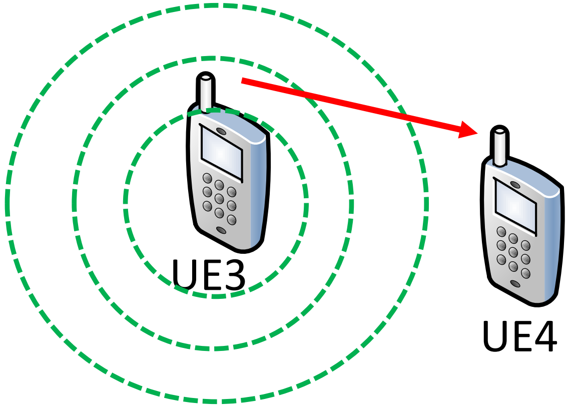
- Why cellular communications and vehicular networks
- SimuLTE background
- Modularity and Interoperability
- Proposed solution

You on 3G
You on 4G
5G

Context

Cellular communications (4G/5G)

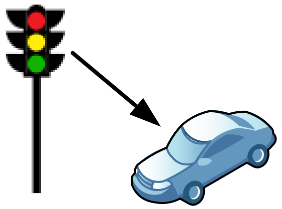
- Ubiquitous coverage
- Support to high speed mobility
- Direct communication (D2D)



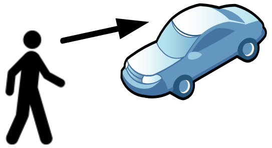
Vehicular networks

- Vehicle to everything (V2X)
- V2V

- V2I

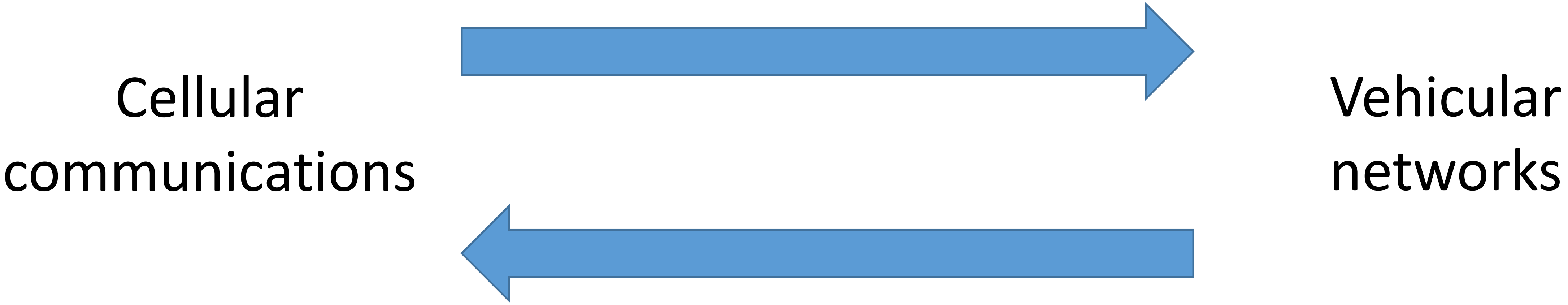


- V2P



Interactions

Enables **fast** and **reliable** communications
Provide multiple **communication paradigms** (D2D vs D2I)



Cellular communications

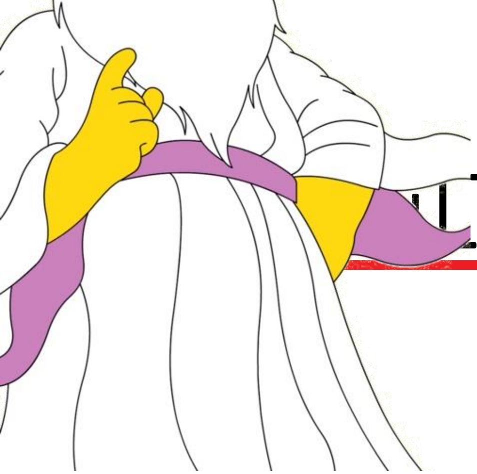
Vehicular networks

Mobility paradigm of users
New challenges:

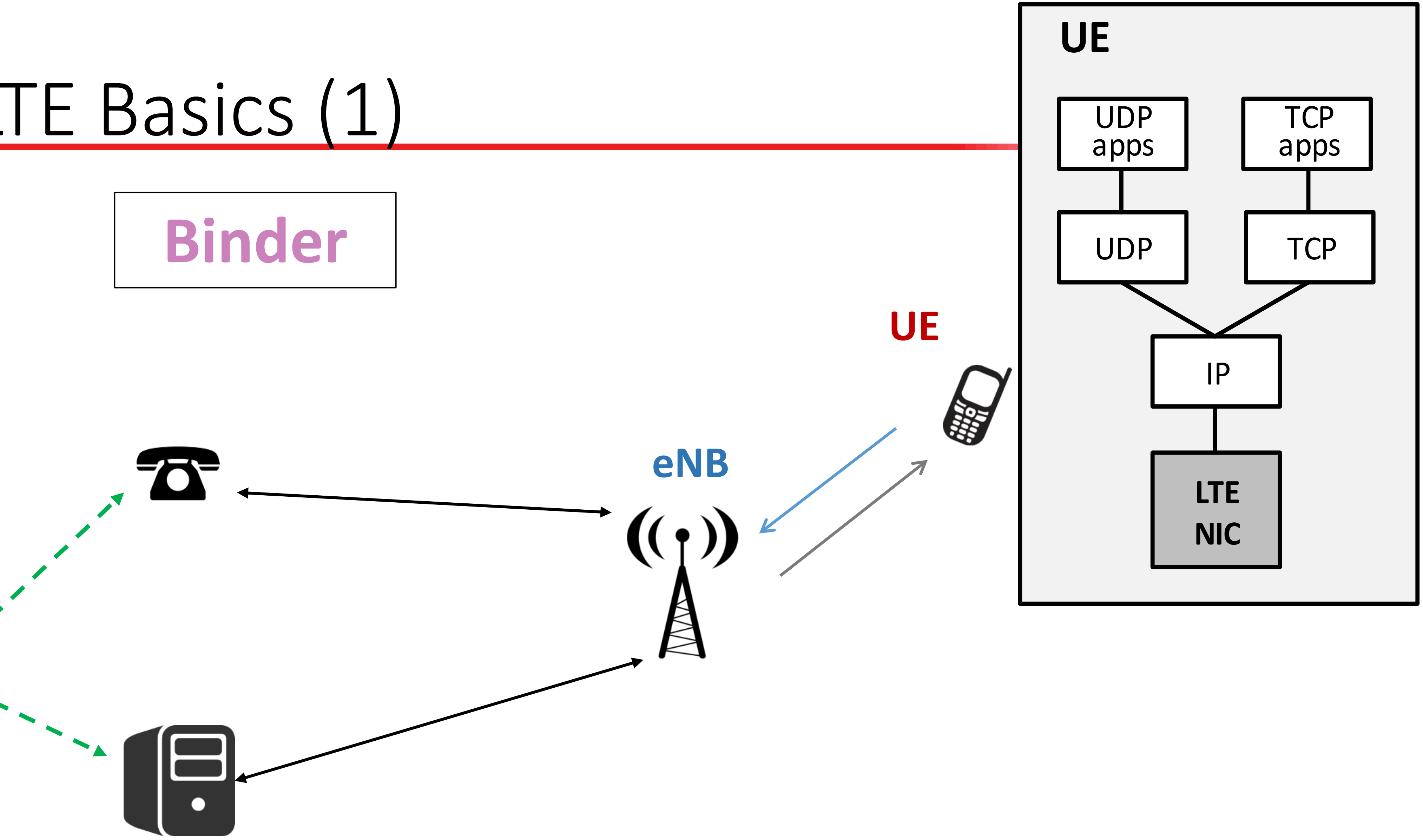
- Bulk handovers
- “Load” mobility
- Stringent **delay constraints**

SimuLTE Basics

- OMNeT-based system-level simulator of LTE networks
- Focused on testing **algorithms for resource scheduling** at large scale
- INET based
- Built as an additional **NIC interface**
- Follow the **evolution** of cellular communications



uLTE Basics (1)



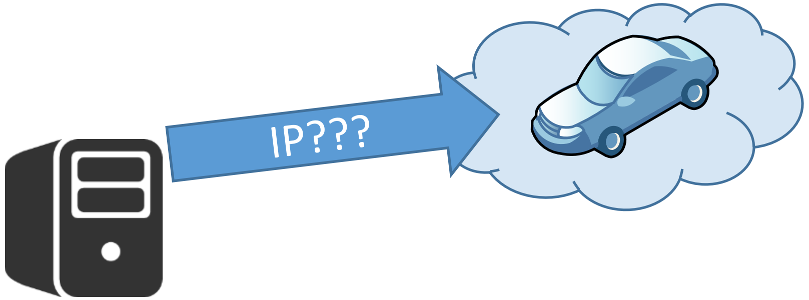
Requirements from Veins

- TraCIMobility

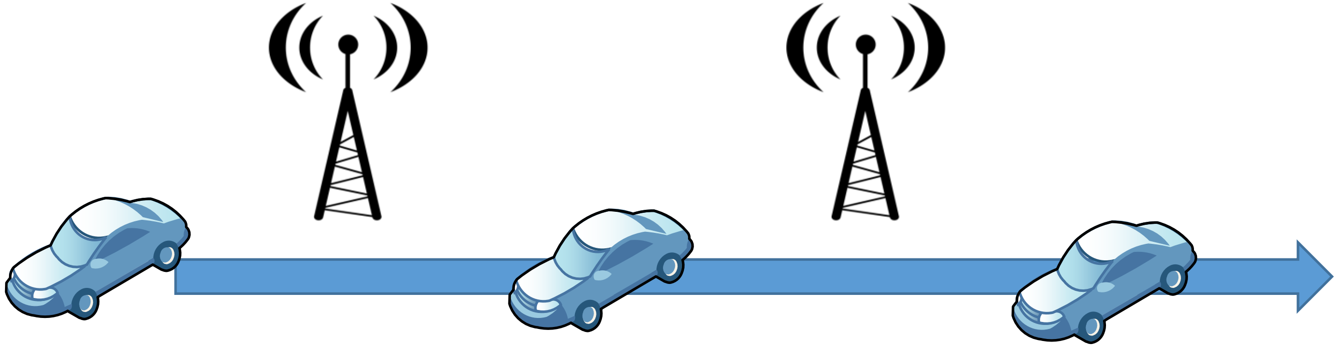
Module responsible for mobility

- Dynamic creation/destruction
  - Module management
  - Addressing

Vehicles can enter/exit the simulation dynamically



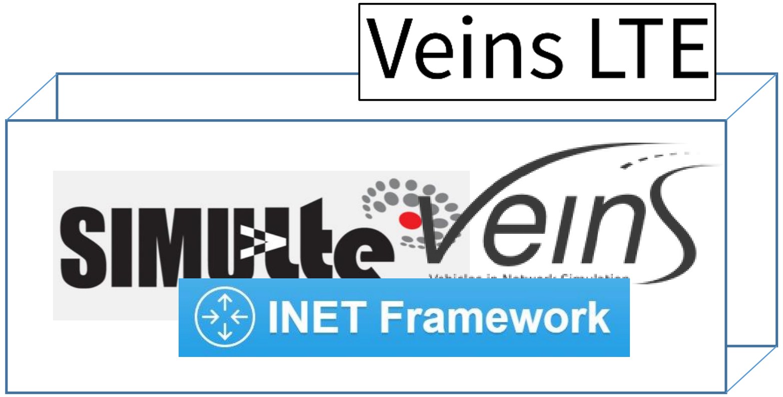
- Handover



Related Work: VeinsLTE

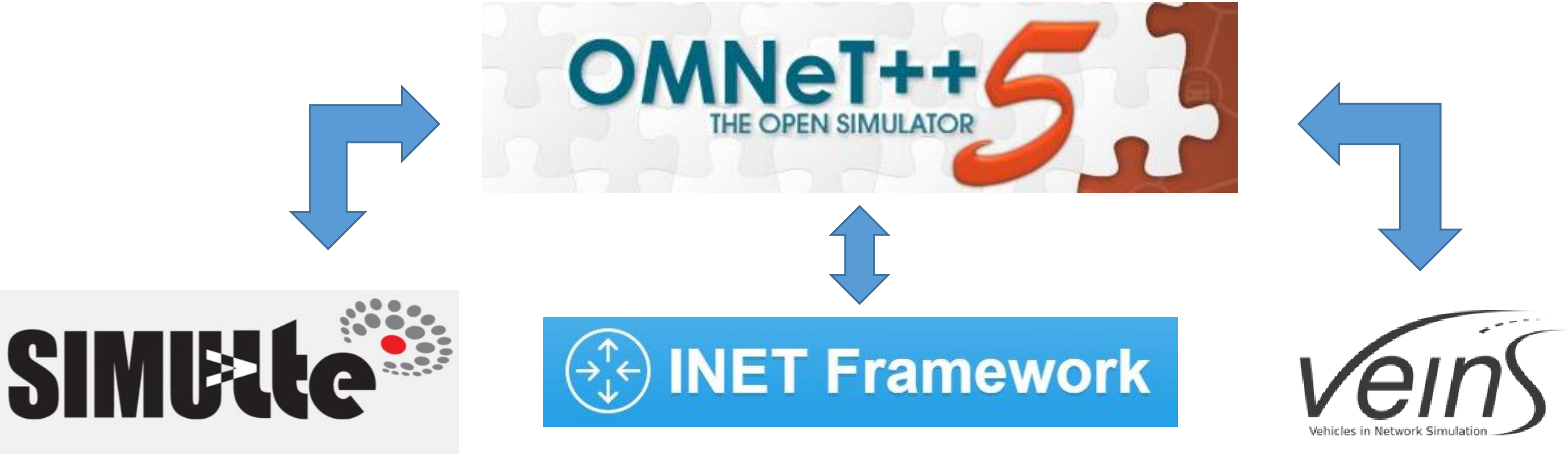
- First attempt to **integrate** SimuLTE and Veins
- Modifications to **both** Veins and SimuLTE
- Veins need to **interact *directly*** with SimuLTE

- <u>Introduces a third framework</u>

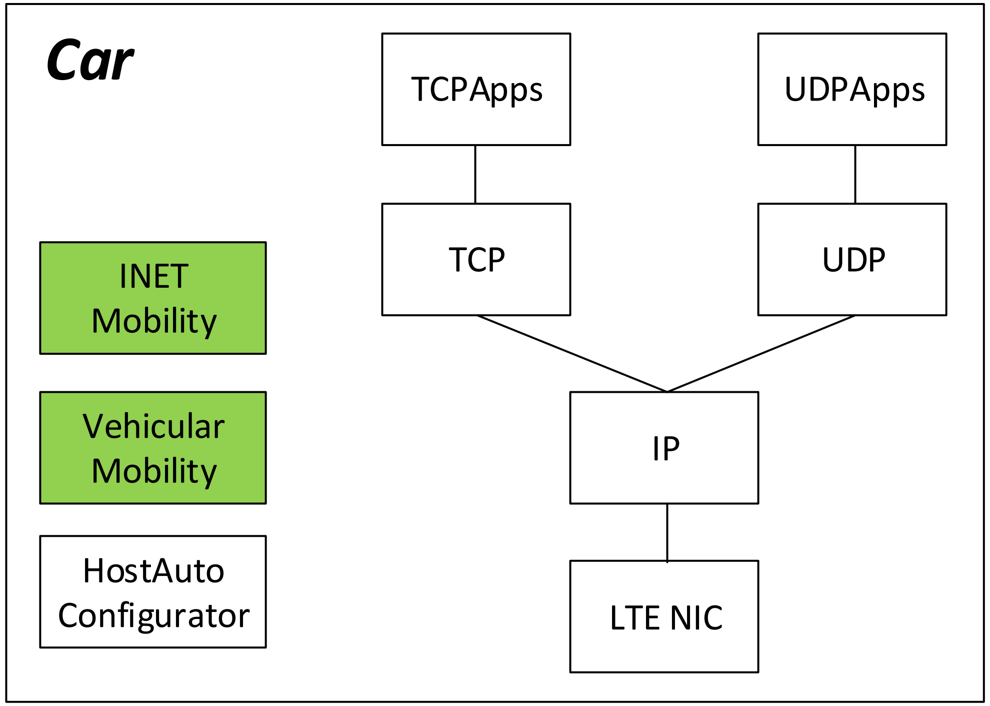


Proposed solution

Modularity and **Interoperability**

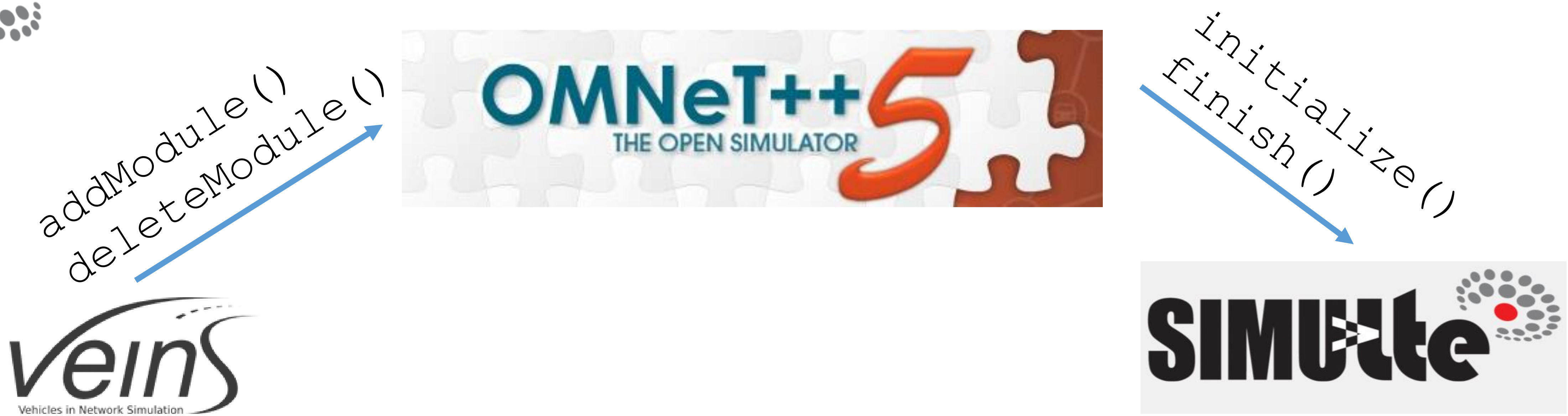


Mobility modules



- Two mobility modules
- Only one active at a time
- Use the *Feature* mechanism
- Interested modules will *register* to the active mobility module

Dynamic creation/destruction: Modules

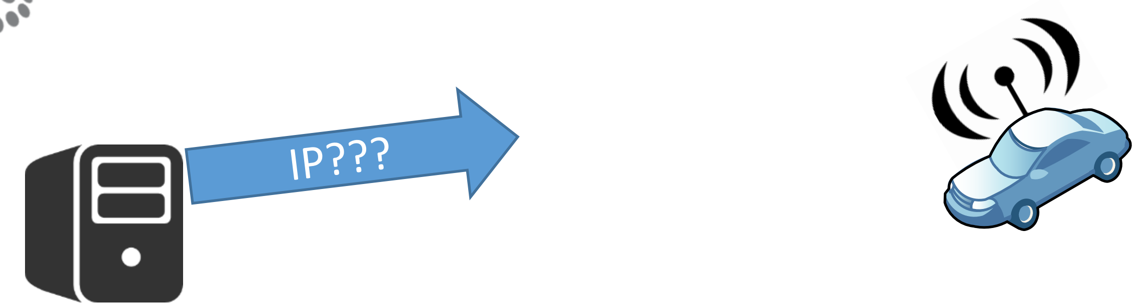


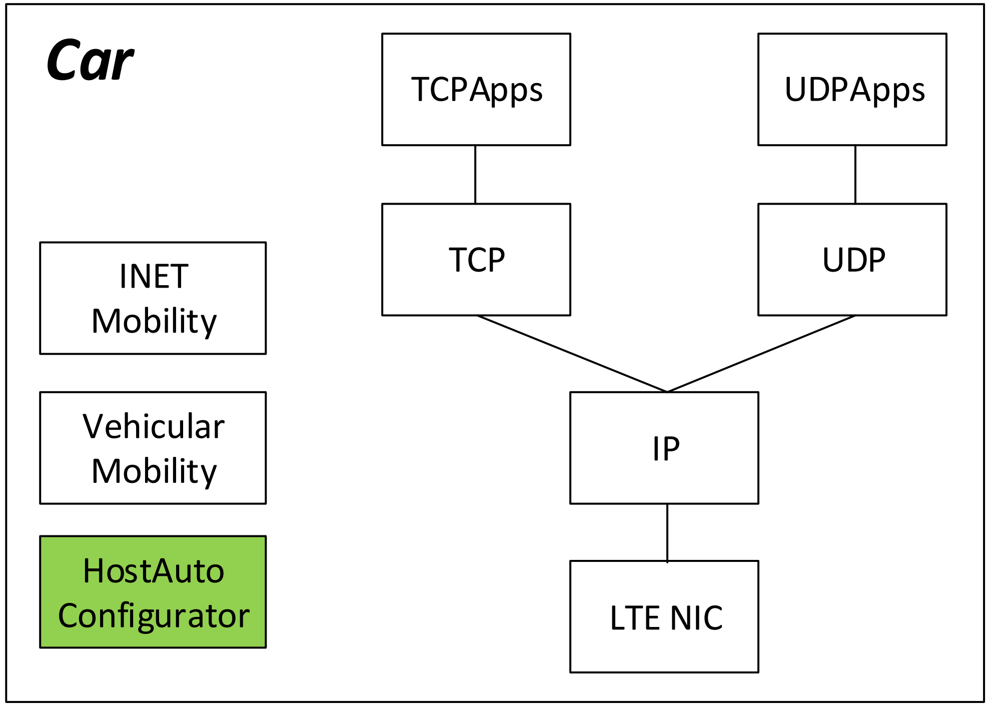
addModule()
deleteModule()

initialize()
finish()

- Initialize and finish functions will interact with the **Binder**, which keeps track of active UEs

Dynamic creation/destruction: Addressing

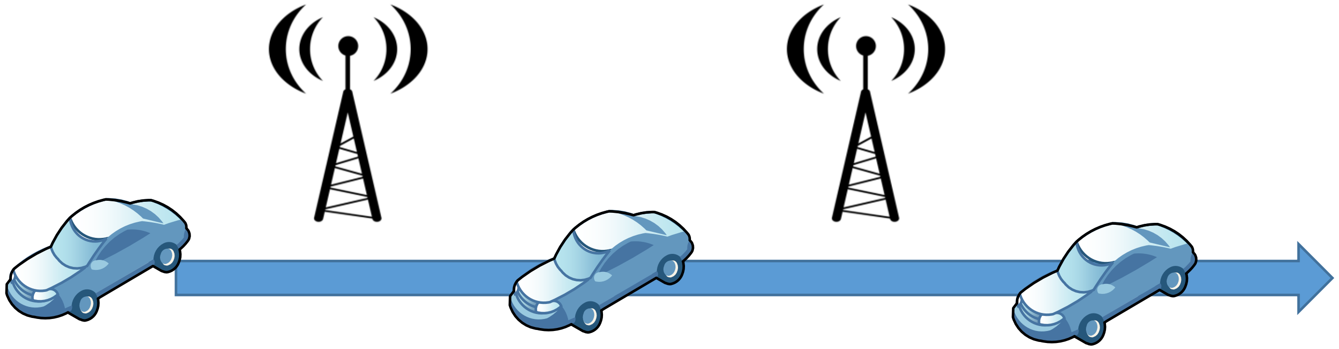




- Designed to minimize modification to applications.
- Node creation: IP autoconfiguration
- Traffic start: check existence
- Node destruction: drop packets

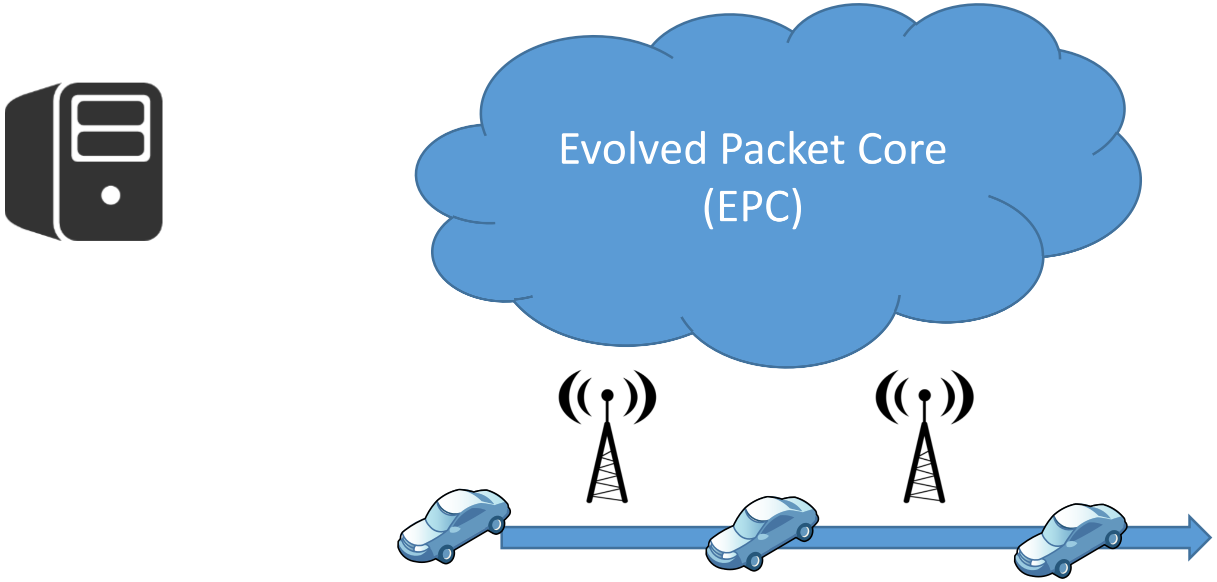
Handover support

- Allows UEs to **change serving cell** -> cell selection
- Implemented using a **best-SINR policy**
- **Initial association** can be either static or handover-like

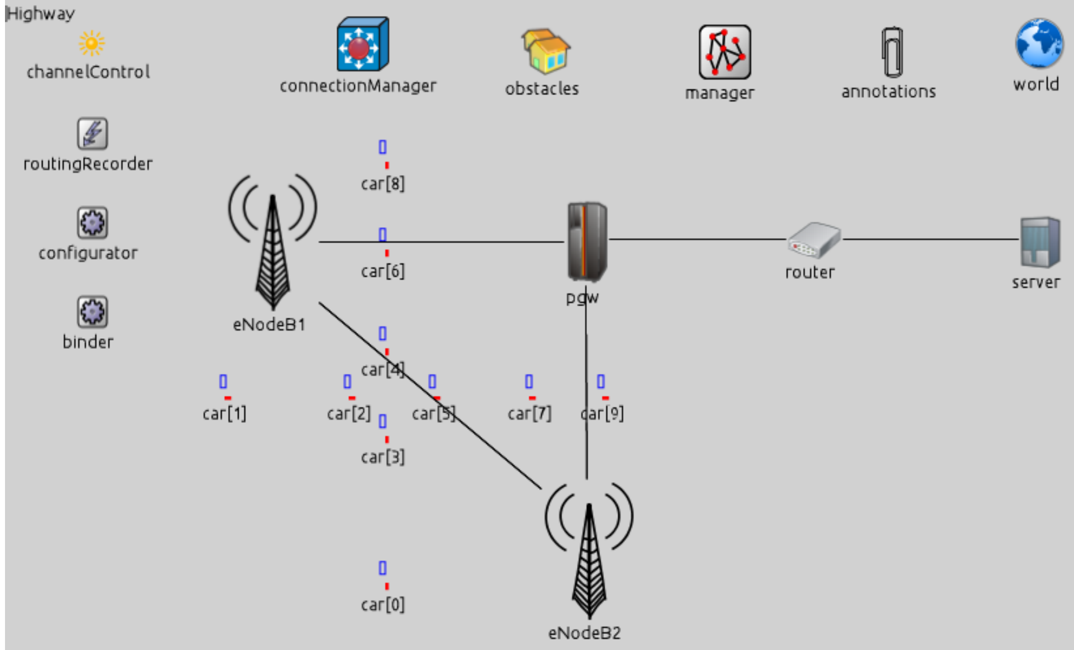


Handover support

- Allows UEs to **change serving cell** -> cell selection
- Implemented using a **best-SINR policy**
- **Initial association** can be either static or handover-like



Exemplary Scenario



```
*.manager.moduleType="lte.corenetwork.nodes.cars.Car"
*.manager.moduleName="car"

*.car[0].vehicularMobility.accidentCount = 1
*.car[0].vehicularMobility.accidentStart = 20s
*.car[0].vehicularMobility.accidentDuration = 30s

**.dynamicCellAssociation = true

*.car[*].masterId = 1
*.car[*].macCellId = 1
```

Further Developments and Case Studies

- Moving towards 5G
  - CRAN deployments
  - Mobile Edge Computing (MEC) applications
  - Include novel protocol aspects
- Implement V2V applications (e.g. platooning)
- Persevere with interoperability (e.g. other mobility models)

Somewhat similar to the concept of **Fog computing**

Conclusions

- Cellular technologies and vehicular networks
- Requirements coming from Veins
- How to support interoperability between Veins and SimuLTE
  - Mobility modules
  - Dynamic creation/destruction
  - Handover support
- Case studies

simulte.com

or

github -> inet -> simulte

Thanks for your attention