A hierarchical approach to design a V2V intersection assistance system

Hycham Aboutaleb
System Engineer, Knowledge Inside
hycham.aboutaleb@k-inside.com

Samuel Boutin
President and CTO,
Knowledge Inside
sbo@k-inside.com

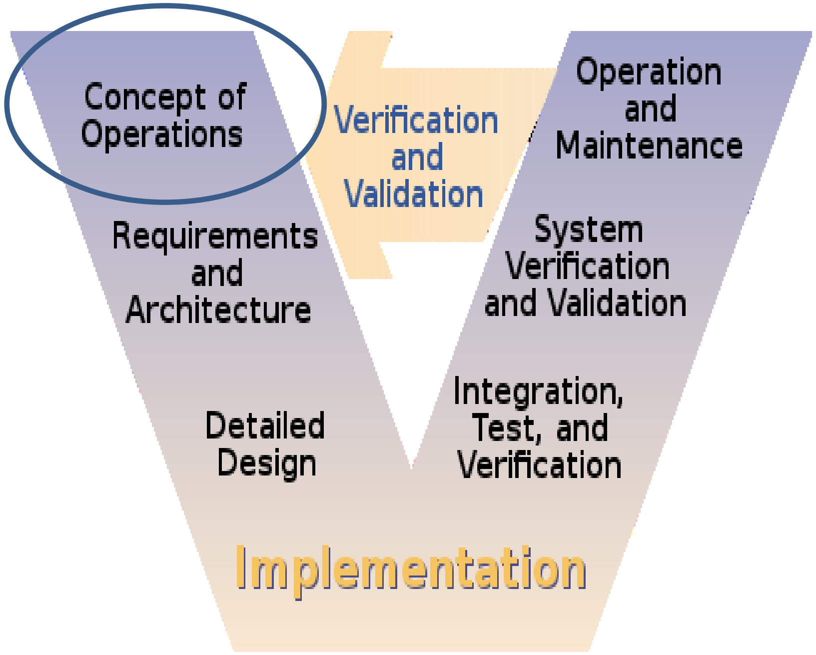
Bruno Monsuez
Associate Professor,
UEI, ENSTA Paristech
bruno.monsuez@ensta.fr

Introduction

- Use cases are often used as thread of events
  ➔ Unrelated to the system design.

- Use cases are described unitarily
  ➔ Incomplete model

- System boundaries are not clear
  ➔ Environment integration in the model?



Context

- Road junctions:
  40-60% of accidents

- Complex problem :
  Huge number of possible scenarios

- Need:
  Manage the combinatorial explosion of possible use cases

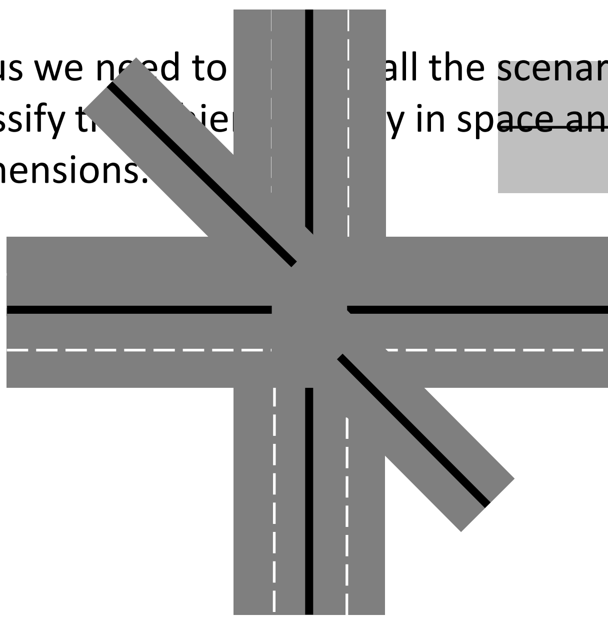
Knowledge Inside
arKItect your system

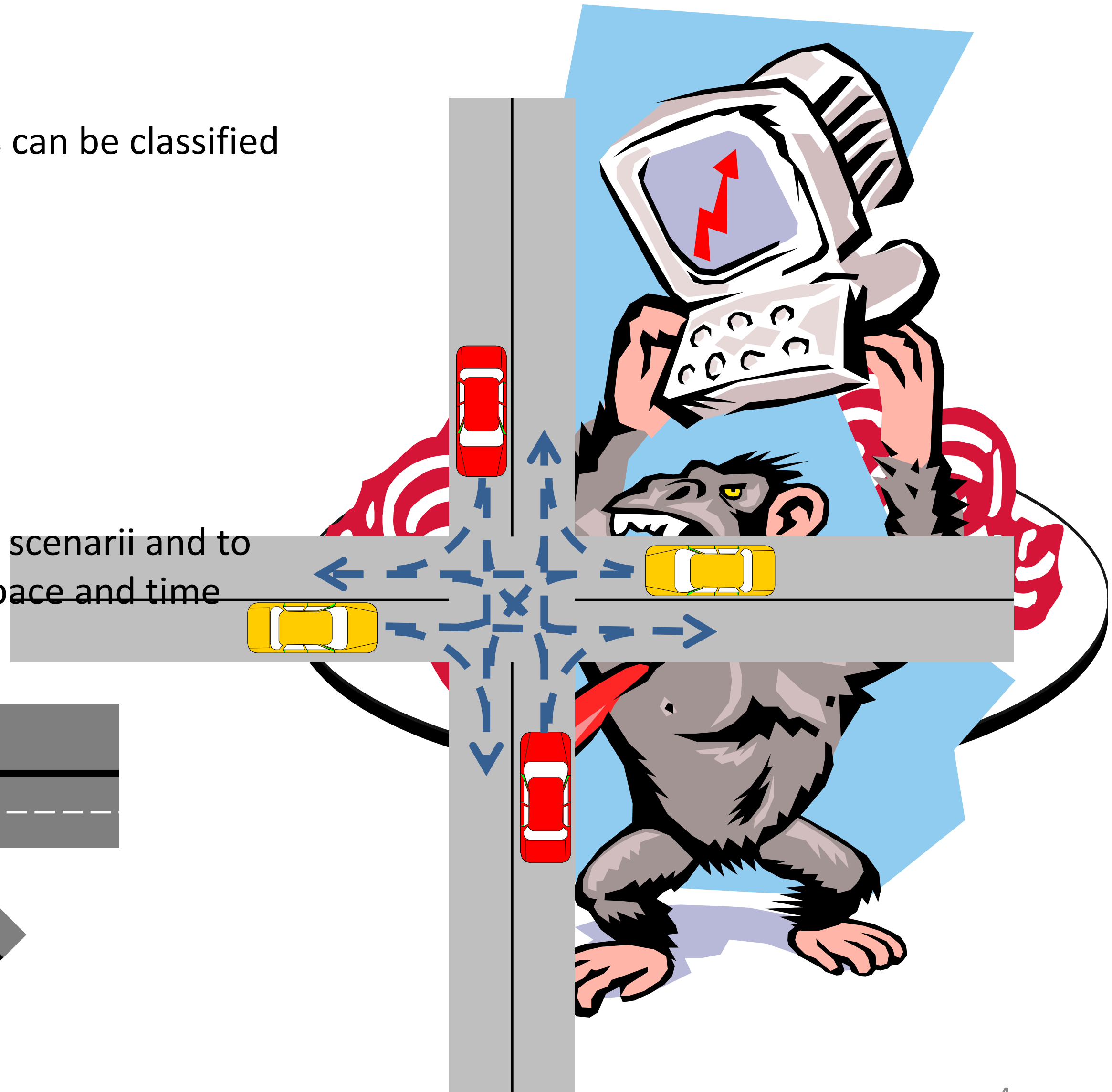
ENSTA ParisTech

Problem Positioning

- The complexity of these systems can be classified into two categories:
  - Complexity in space
  - Complexity in time

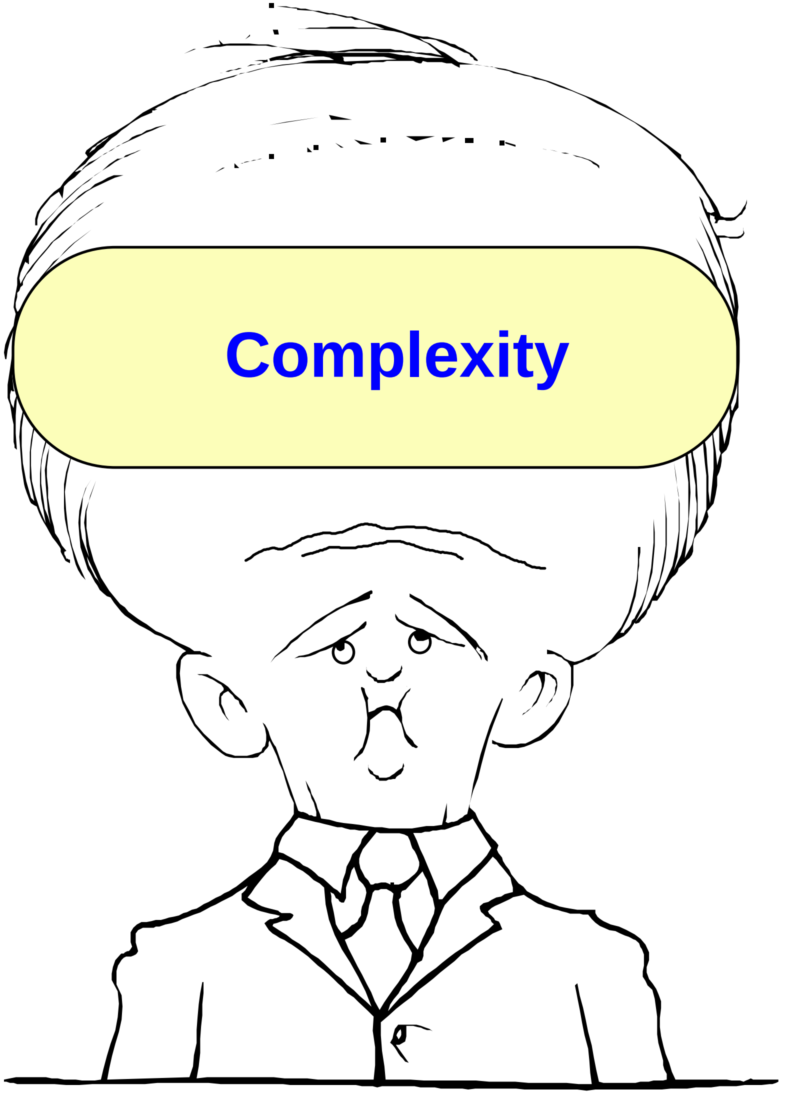
- Thus we n ed to all the scenarii and to classify t ie y in space and time dimensions.





Our Methodology

- To deal with the complexity of the problem we need several reductions:
  - **Structural Reduction**
    - Identify the system
    - Manage topologies diversity
  - **Dynamical Reduction**
    - Reduce to involved elements
    - Reduce using symmetry
  - **Behavioral Reduction**
    - Reduce to vehicles behavior
    - Identify scenario constraints
  - **Decisional Reduction**
    - Identify decisions for the scenario
    - Identify actions for the scenario



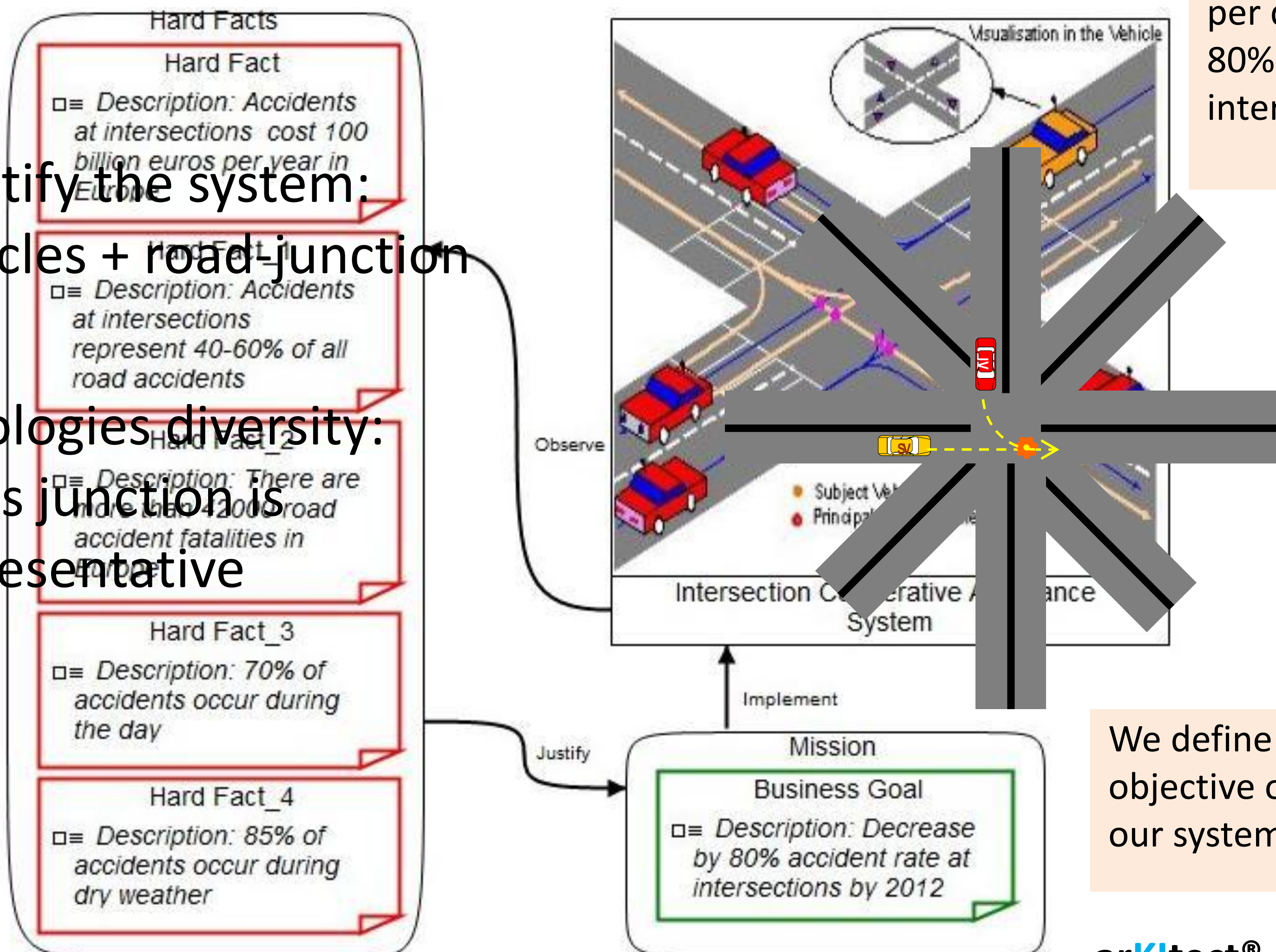
Structural Reduction

Top Level: Environment

We justify the need to achieve this goal.

Road junction with one lane per direction: 80% of intersections

- Identify the system: vehicles + road-junction
- Topologies diversity: Cross junction is representative



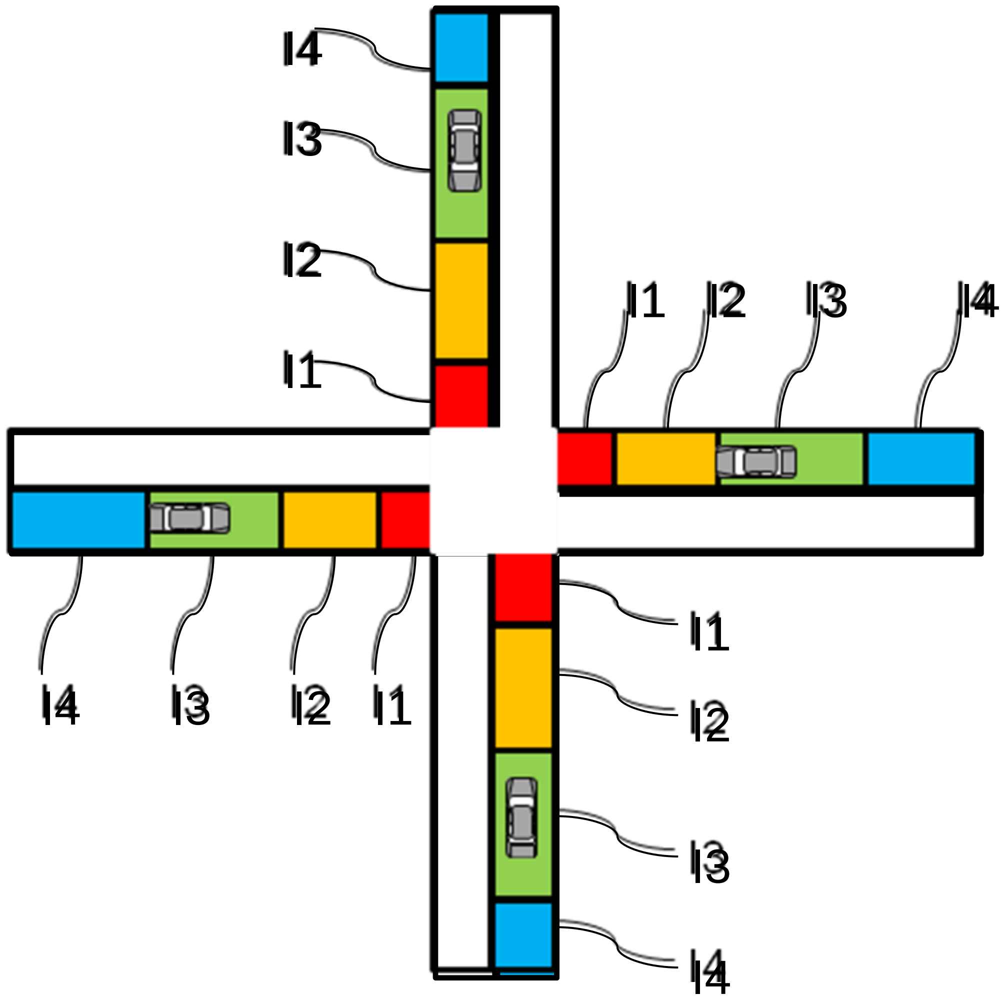
We define the objective of our system

arKItect®

Dynamic Reduction (1)

First Level: Selecting the Vehicles

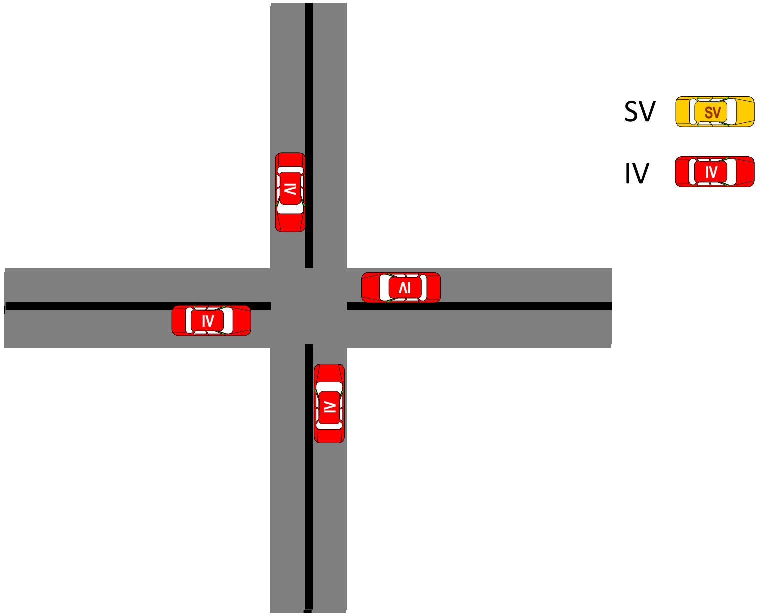
- Vehicles that have the same interval of time-to-collision might collide.
- Matching vehicles will communicate together only.
- Complexity is reduced to a maximum of 4 vehicles



Dynamic Reduction (2)

Second Level: Selecting pairs of vehicles (1)

- We take the perspective of a vehicle arriving at the intersection and call it *Subject Vehicle* (SV).
- All other vehicles are considered *Intruder Vehicles* (IV).
- This choice is completely arbitrary and in no way determines the priority of each vehicle.

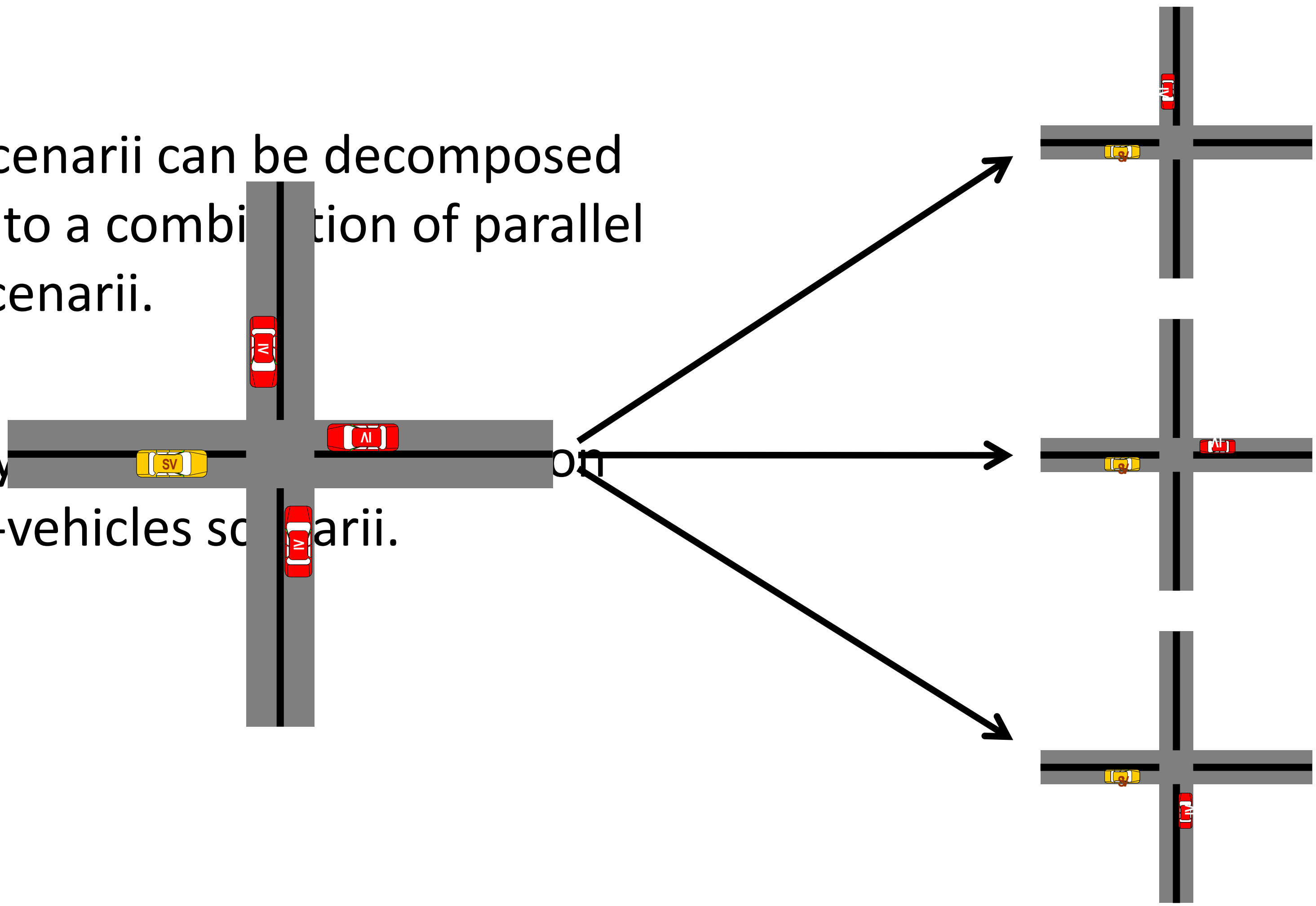


Dynamic Reduction (2)

Second Level: Selecting pairs of vehicles (2)

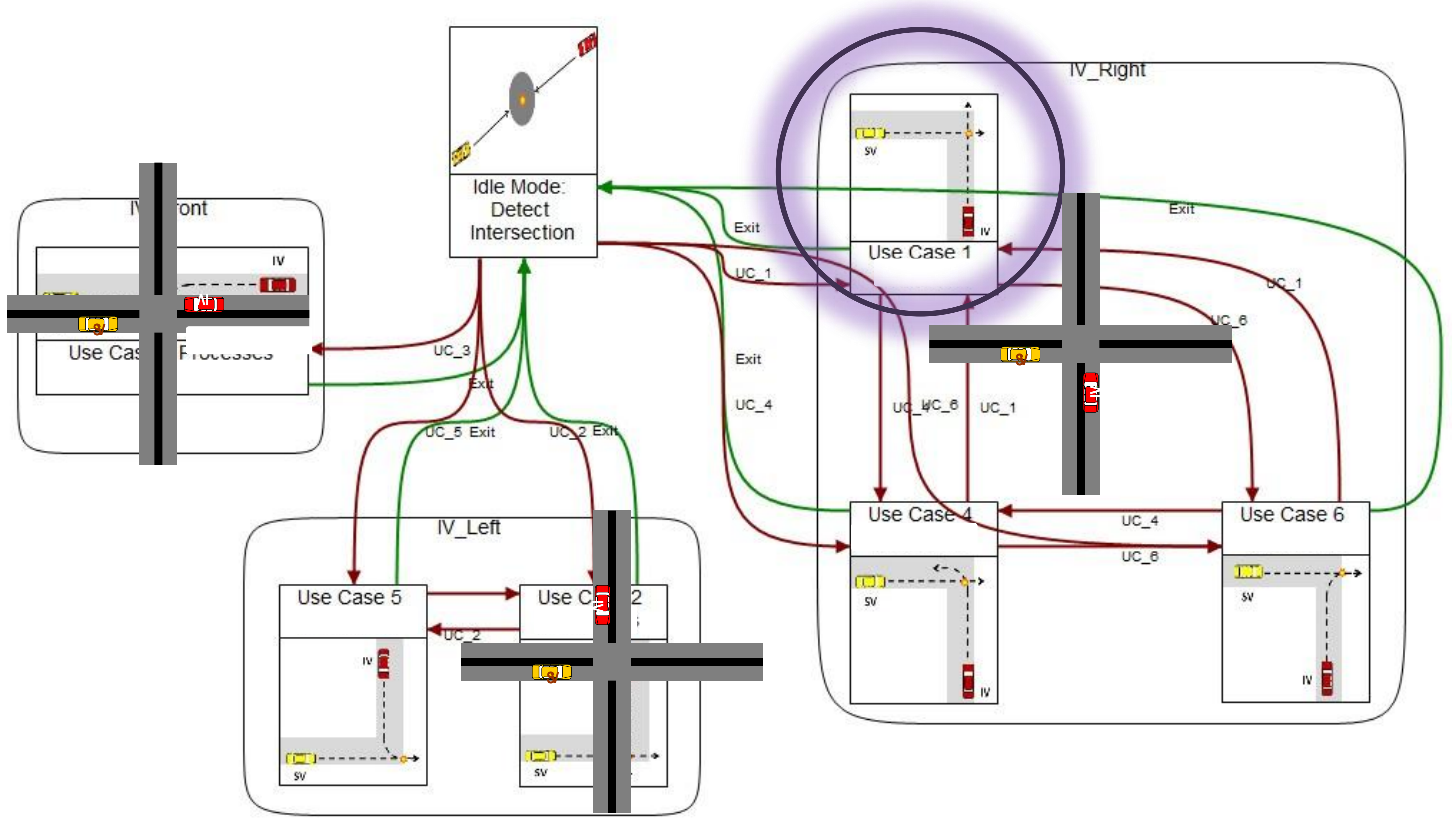
- Scenarii can be decomposed into a combi tion of parallel scenarii.

- By on 2-vehicles sc arii.



Behavioral Reduction (1)

Third Level: Identifying all the Scenarii



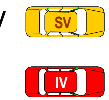
arKItect®

Behavioral Reduction (2)

Fourth Level: Managing priorities

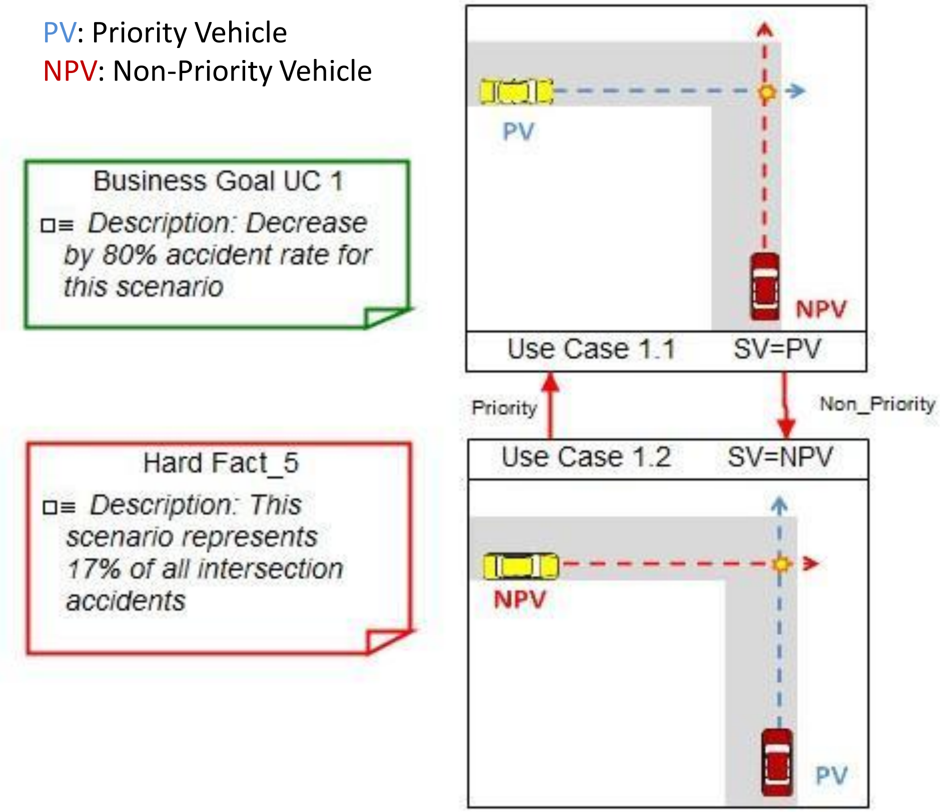
SV

IV



- For each use case, manage the transition of the two vehicles by priority for each.
- The change of priority implies the passage from one state to another.

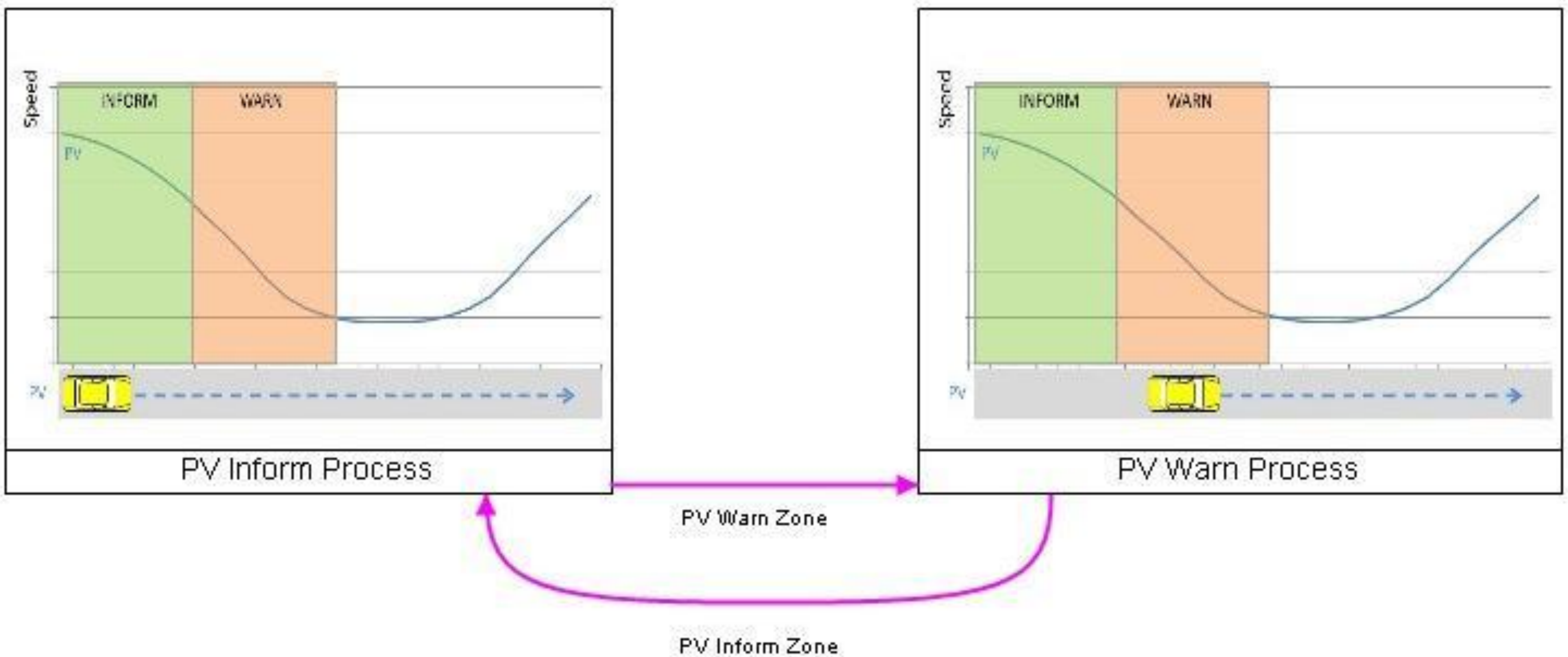
PV: Priority Vehicle
NPV: Non-Priority Vehicle



arKItect®

Decisional Reduction

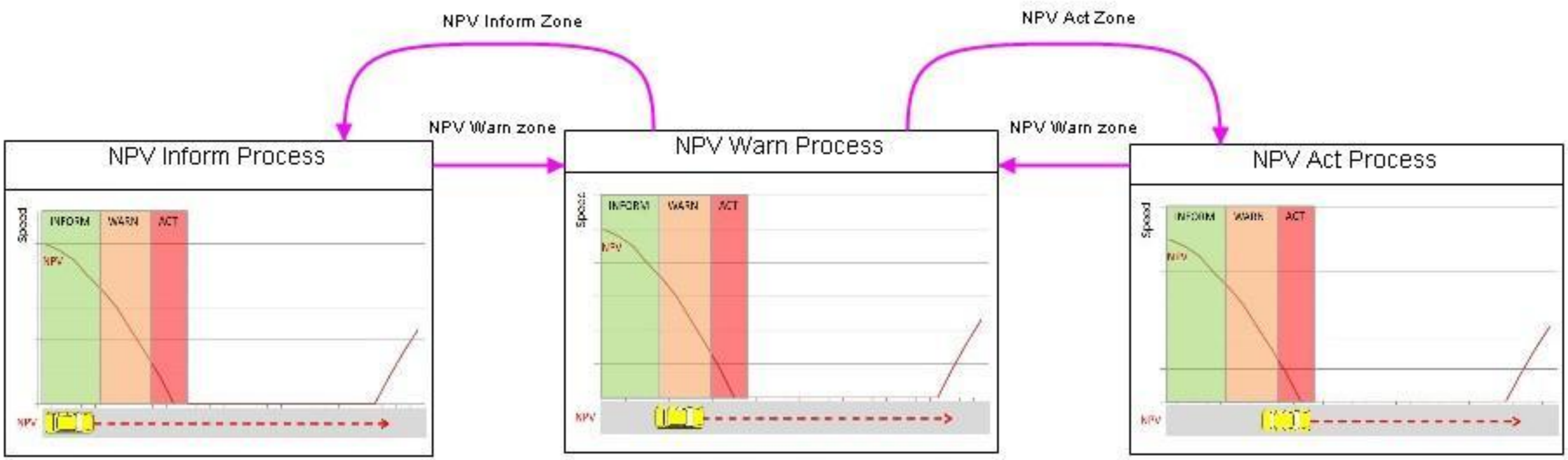
Fifth Level: Acting and deciding (1)



arKItect®

Decisional Reduction

Fifth Level: Acting and deciding (2)

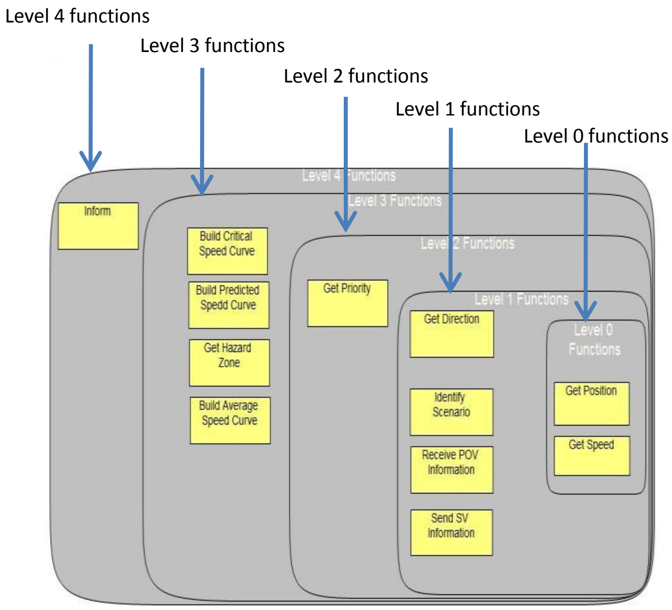


arKItect®

Advantages of our approach

- Strong and intuitive link between use cases and functional architecture.

- Graphical traceability to check coherence.



arKItect®

Conclusion

- Despite the fact that use cases in themselves are quite intuitive, the process around them is a much bigger.
- The top-down approach that we followed leads to greater efficiency in complex tasks.
- The model-based design is a result of this decomposition.
- It is expected that the resulting system will be applicable to a wider range of accident scenarios.

Future Works

- Simulate (MIL, SIL): Verify the model through simulation, which needs to model and simulate a user behavior and his reactions to MMI messages in each situation.

- Code generation: From the detailed model and based on the use cases, an automatic code generation can be performed

- Estimate software design cost: an analysis for each elementary function can be performed. The cost should be assessed according to the hierarchical decomposition (analysis, design test for each level).

Thanks for your Attention !!

QUESTIONS?