A Standard Driven Software Architecture for Fully Autonomous Vehicles

WASA 2018

Alexandru C. Serban[1,2]
Erik Poll[1]
Joost Visser[1,2]

[1]Radboud University, Nijmegen

[2]Software Improvement Group, Amsterdam

The Netherlands

01.05.2018

Contents

What is desired? Fully autonomous vehicles

Definition (Functional Software Architecture)

A specification of intended *functions* and necessary *interactions* in order to achieve desired behaviors.

Why? Propositions

Drawbacks in literature:

Why? Propositions

Drawbacks in literature:

1. After-math of building a prototype

Why? Propositions

Drawbacks in literature:

1. After-math of building a prototype
2. Requirements are hard to trace w.r.t functional components

Why? Propositions

Drawbacks in literature:

1. After-math of building a prototype
2. Requirements are hard to trace w.r.t functional components
3. Few components group most functionality

Why? Propositions

Drawbacks in literature:

1. After-math of building a prototype
2. Requirements are hard to trace w.r.t functional components
3. Few components group most functionality
4. No rationale for decision making

How?

A better approach?

How?

A better approach?

1. Clear (standard) requirements

How?

A better approach?

1. Clear (standard) requirements
2. Functional decomposition w.r.t to the automotive software development life-cycle

How?

A better approach?

1. Clear (standard) requirements
2. Functional decomposition w.r.t to the automotive software development life-cycle
3. Rationale for decision making

Higher level goals

The autonomous driving *demonstrators* involves some sort of *perception* and higher *intelligence* plugged on top of a base vehicle platform which usually incorporates computerized *control* of functions like propulsion and braking [1].

Higher level goals

The autonomous driving *demonstrators* involves some sort of *perception* and higher *intelligence* plugged on top of a base vehicle platform which usually incorporates computerized *control* of functions like propulsion and braking [1].

Some natural thoughts about *intelligence*:

Higher level goals

The autonomous driving *demonstrators* involves some sort of *perception* and higher *intelligence* plugged on top of a base vehicle platform which usually incorporates computerized *control* of functions like propulsion and braking [1].

Some natural thoughts about *intelligence*:

1. Where do we *plug* in the intelligence?

Higher level goals

The autonomous driving *demonstrators* involves some sort of *perception* and higher *intelligence* plugged on top of a base vehicle platform which usually incorporates computerized *control* of functions like propulsion and braking [1].

Some natural thoughts about *intelligence*:

1. Where do we *plug* in the intelligence?
2. How do we *improve* our intelligence?

Higher level goals

The autonomous driving *demonstrators* involves some sort of *perception* and higher *intelligence* plugged on top of a base vehicle platform which usually incorporates computerized *control* of functions like propulsion and braking [1].

Some natural thoughts about *intelligence*:

1. Where do we *plug* in the intelligence?
2. How do we *improve* our intelligence?
3. How can we *distribute* as much as possible classic control tasks?

Further decomposition of perception and intelligence

Where do we start?

Further decomposition of perception and intelligence

Where do we start?

- Search for a set of standard requirements

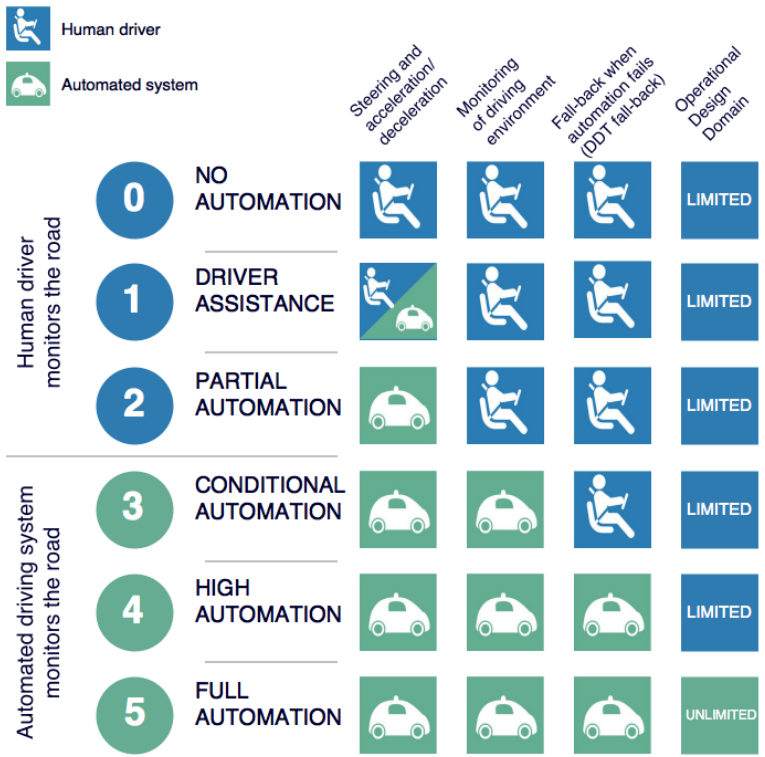
Further decomposition of perception and intelligence

Where do we start?

- Search for a set of standard requirements
- SAE J3016

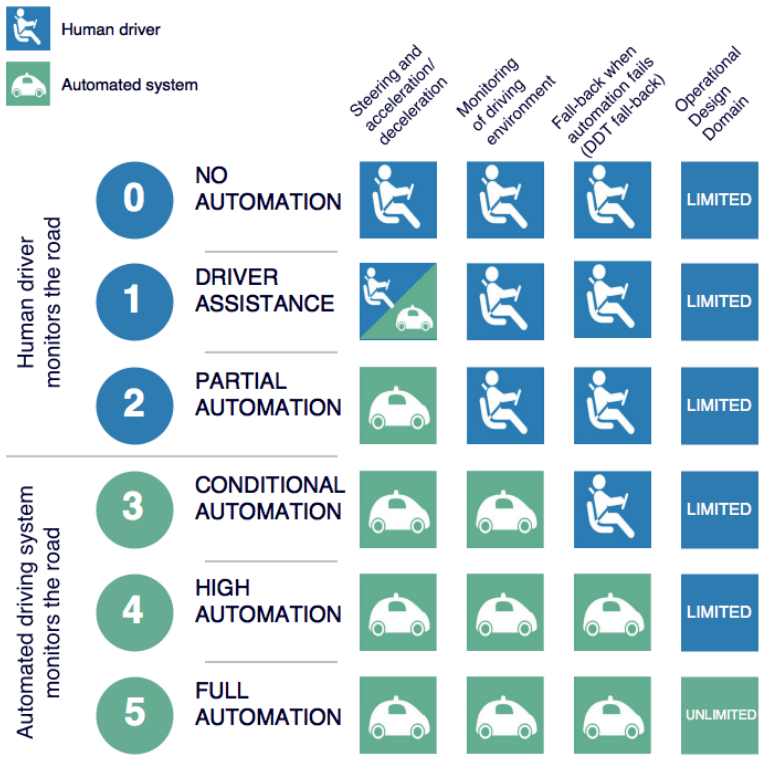
From no automation to full automation

- [*SAEJ*3016] . . . describes driving automation features and the underlying principles used to *evolve* from *none* to *full* driving automation.



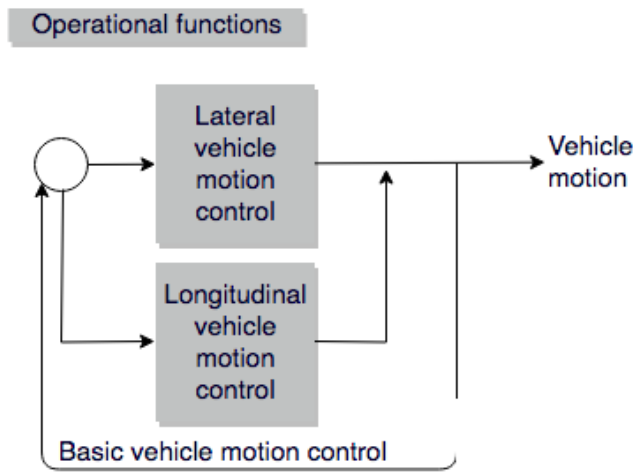
From no automation to full automation

- [*SAEJ*3016] . . . describes driving automation features and the underlying principles used to *evolve* from *none* to *full* driving automation.
- It is *not* meant as a *blueprint*, *but* at the moment it is the only *standard functional description* for building autonomous vehicles.



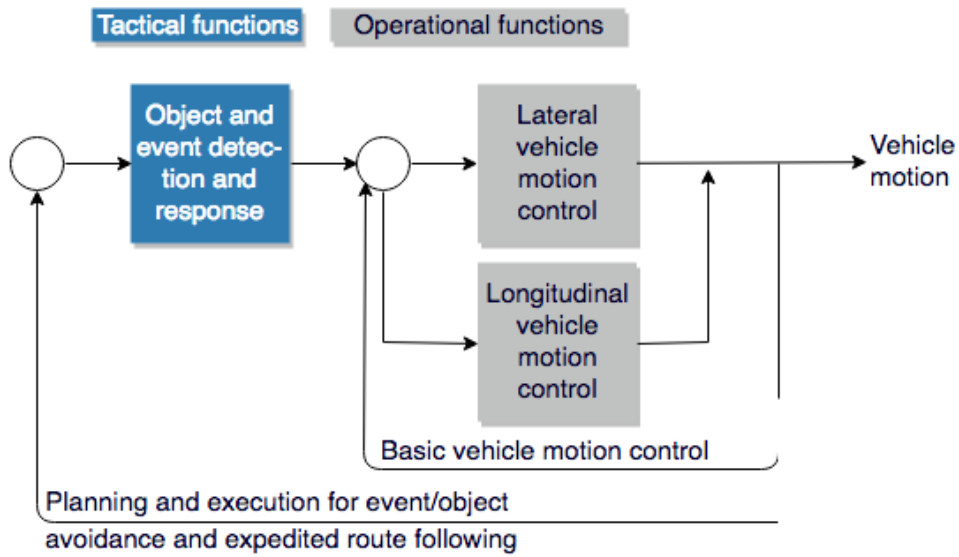
Requirements

1. **Operational** - basic vehicle control



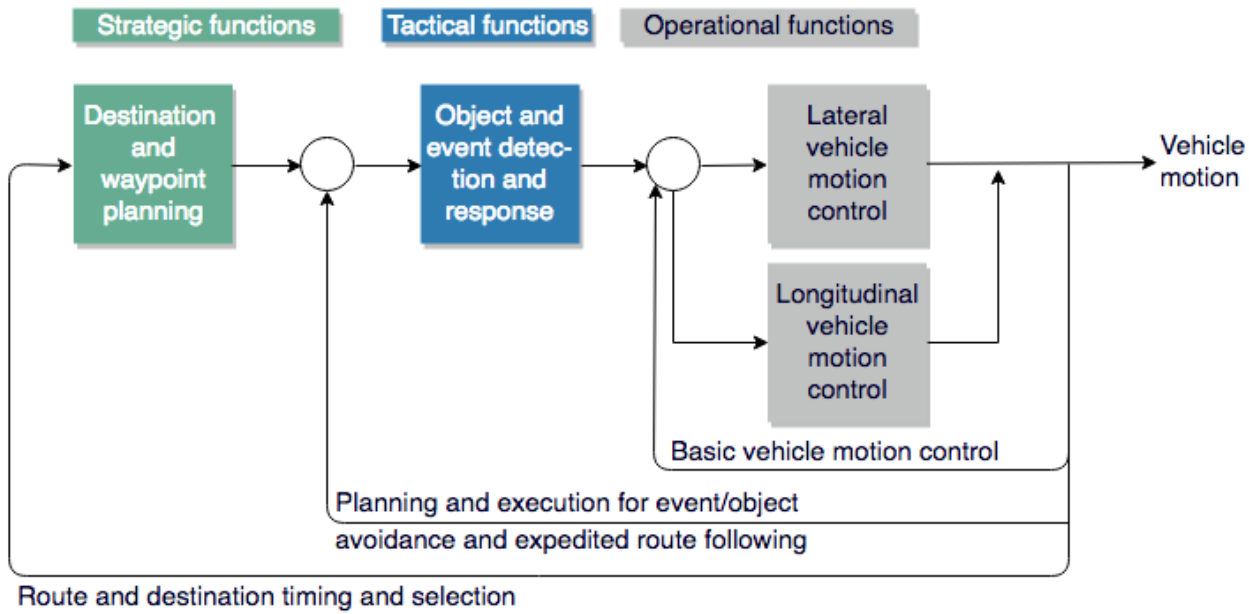
Requirements

1. **Operational** - basic vehicle control
2. **Tactical** - planning and execution for event/object avoidance and expedited route following



Requirements

1. **Operational** - basic vehicle control
2. **Tactical** - planning and execution for event/object avoidance and expedited route following
3. **Strategic** - destination and general route planning



Rationale

Remark

Architecture design for autonomous vehicles is analogous to the design of a real-time, intelligent, control system (a.k.a a robot).

Rationale

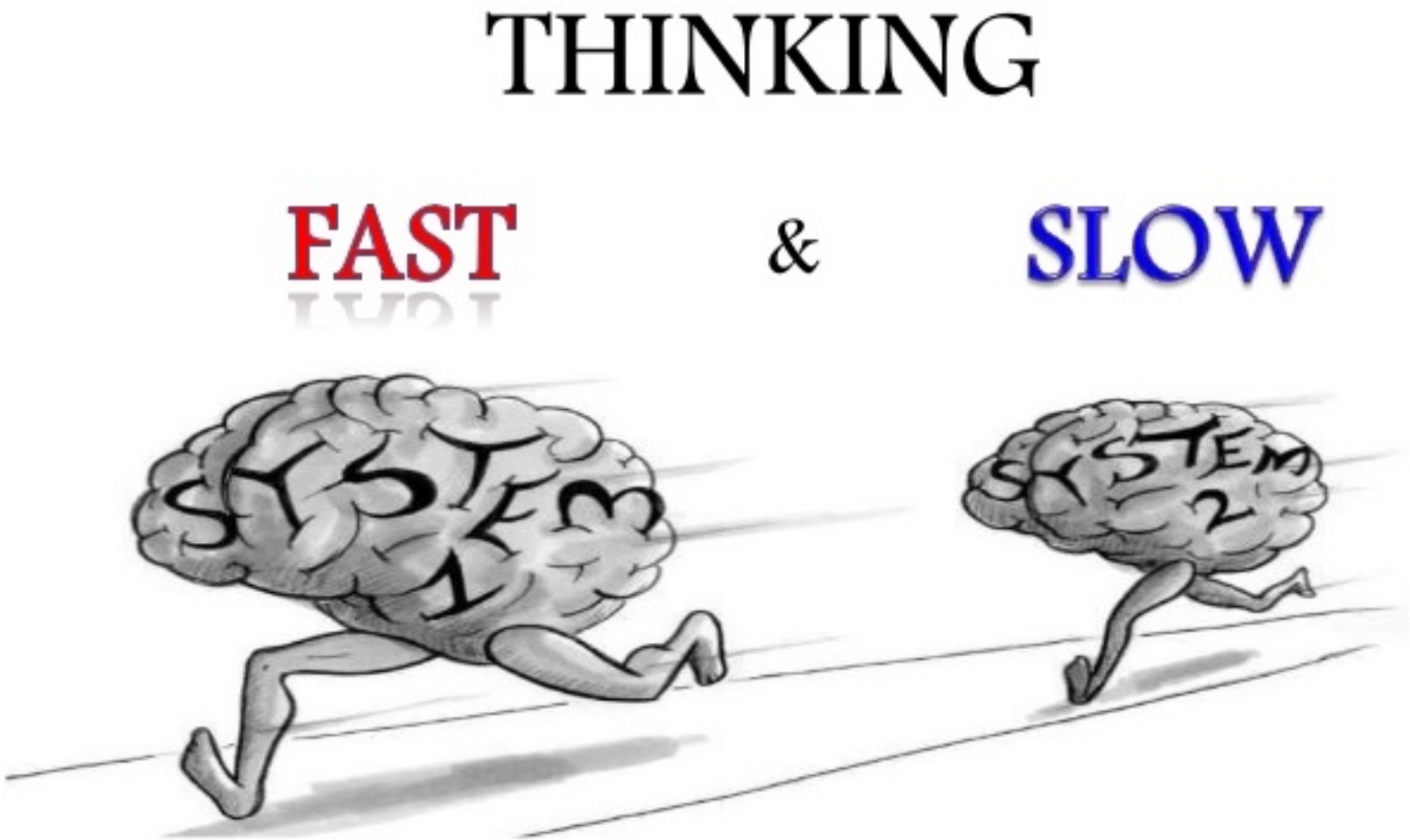
Remark

Architecture design for autonomous vehicles is analogous to the design of a real-time, intelligent, control system (a.k.a a robot).

Existing AI or Robotics architectures:

- Reactive
- Deliberative

How about both?[2]



World Modeling

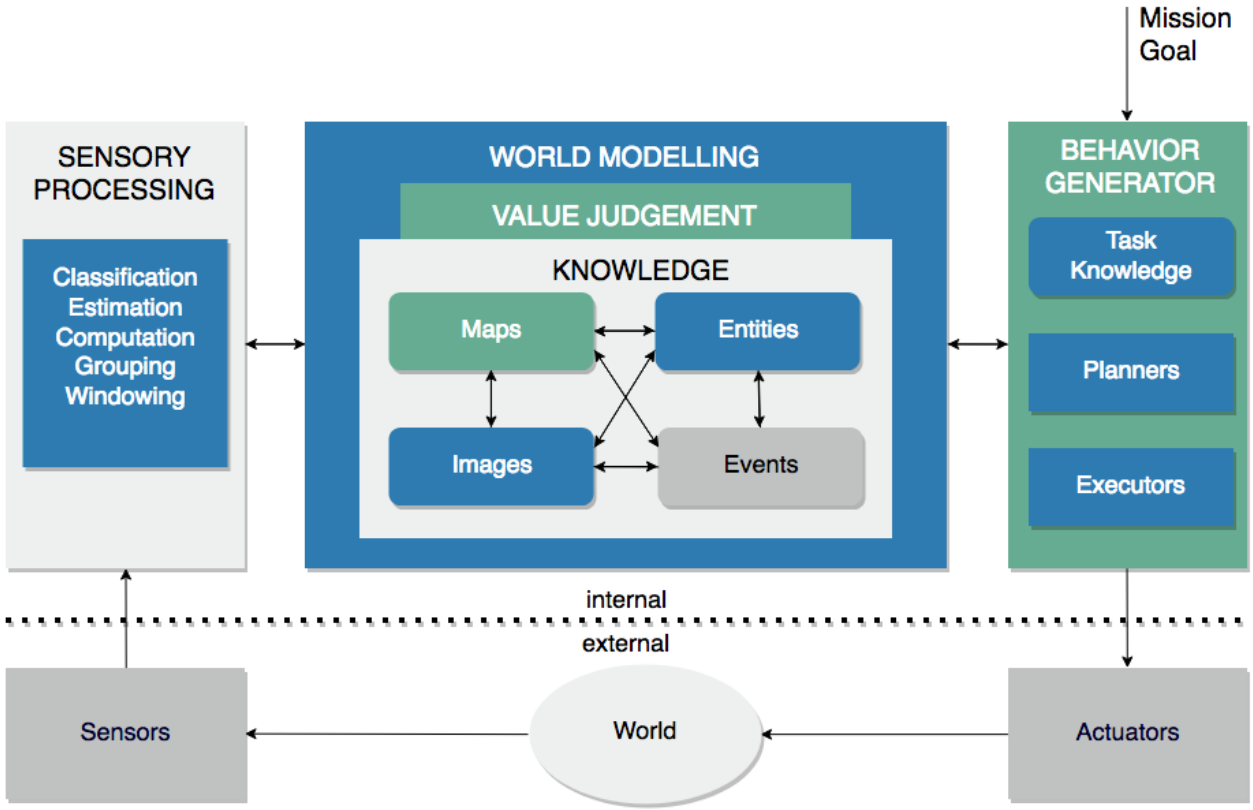
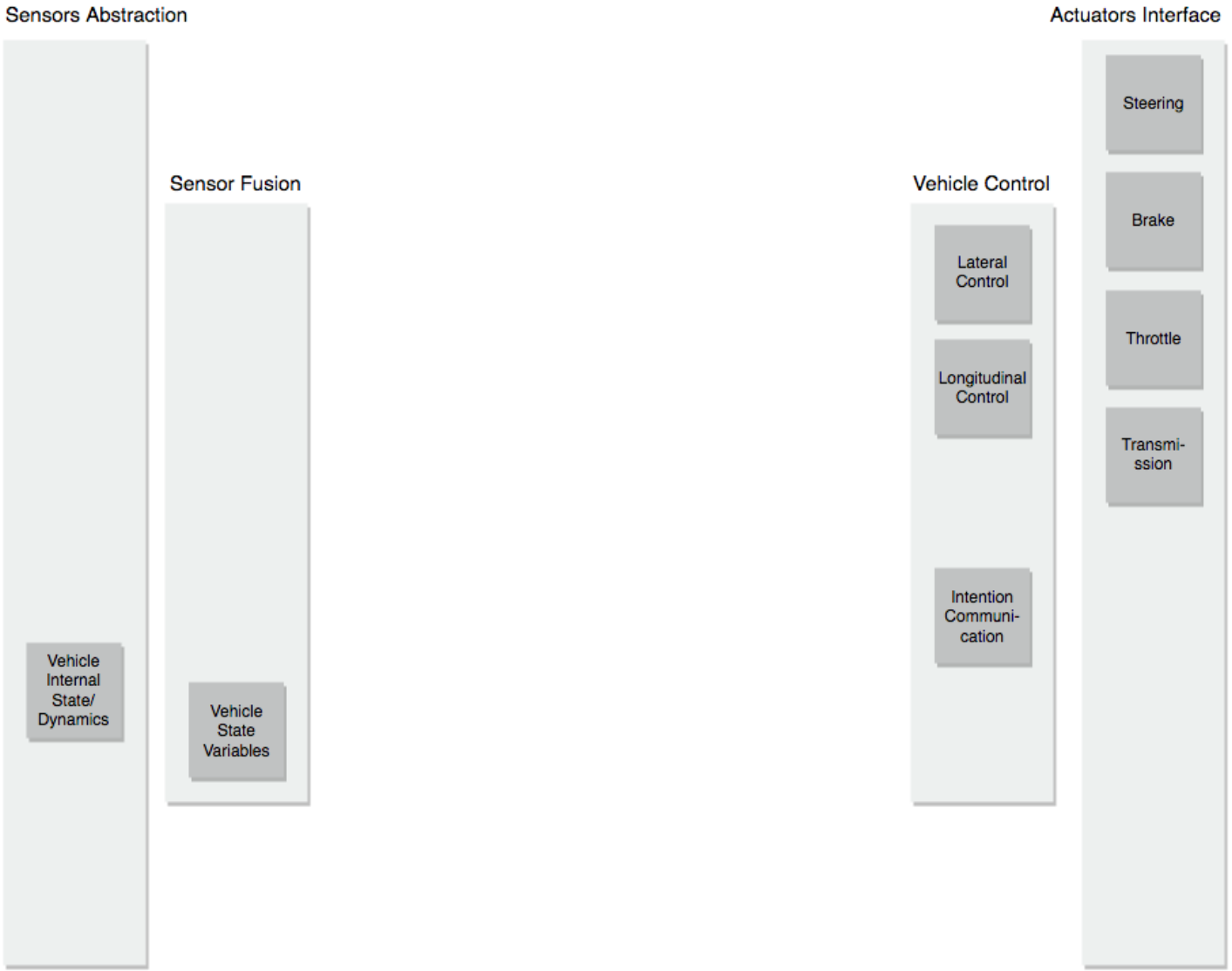


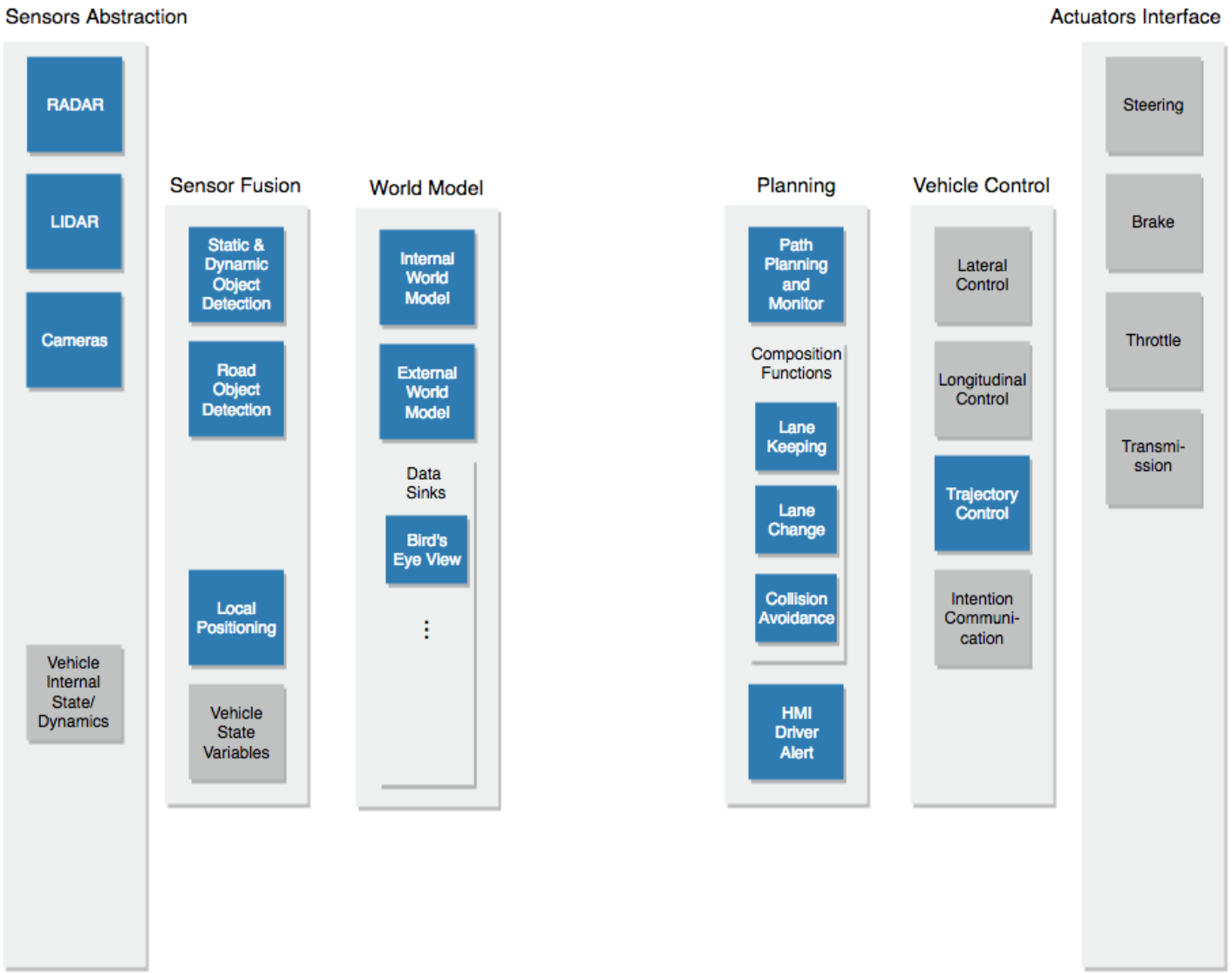
Figure: NIST RCS reference architecture.

A cocktail of all notions introduced so far.

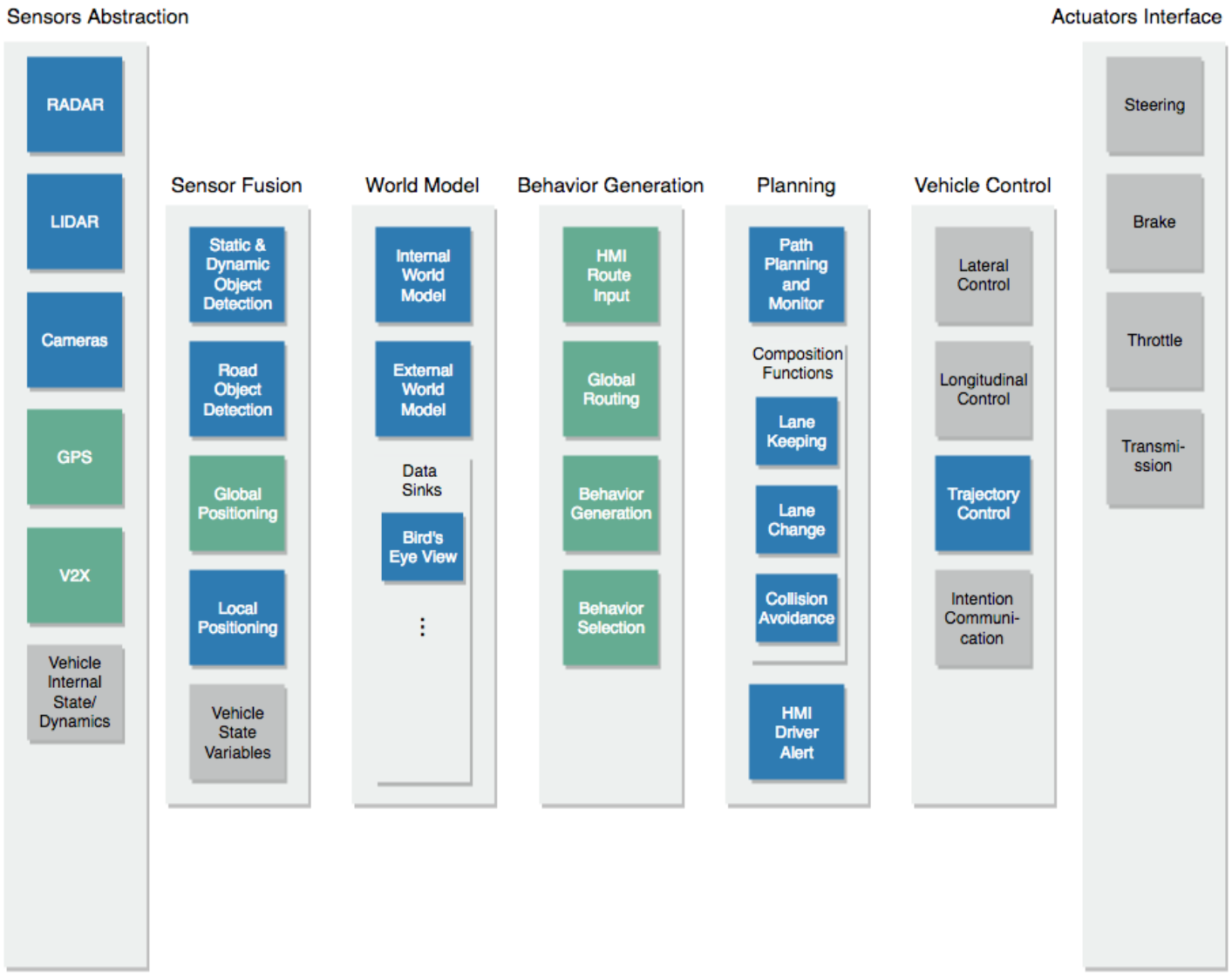
Basic operational functions



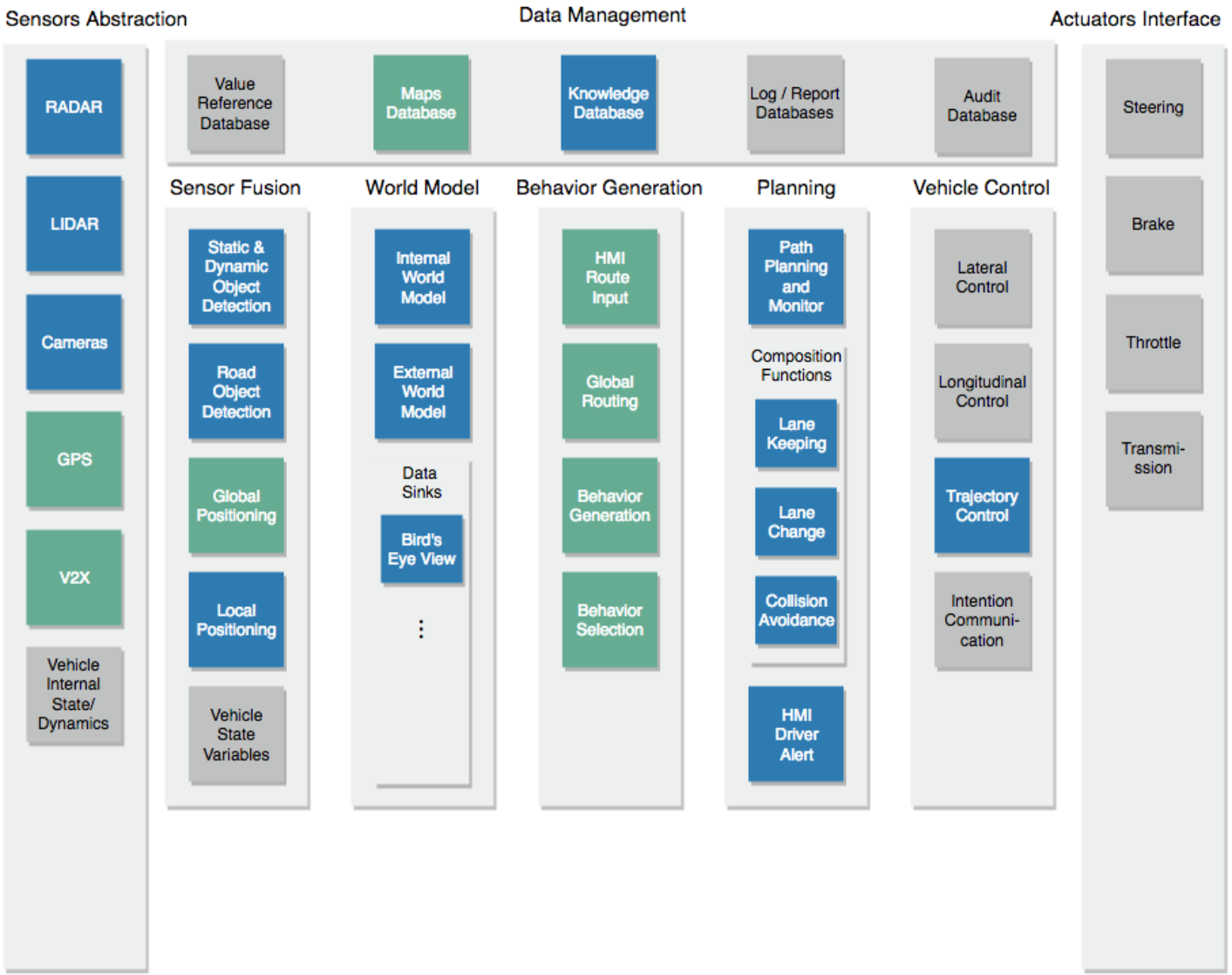
Adding tactical functions



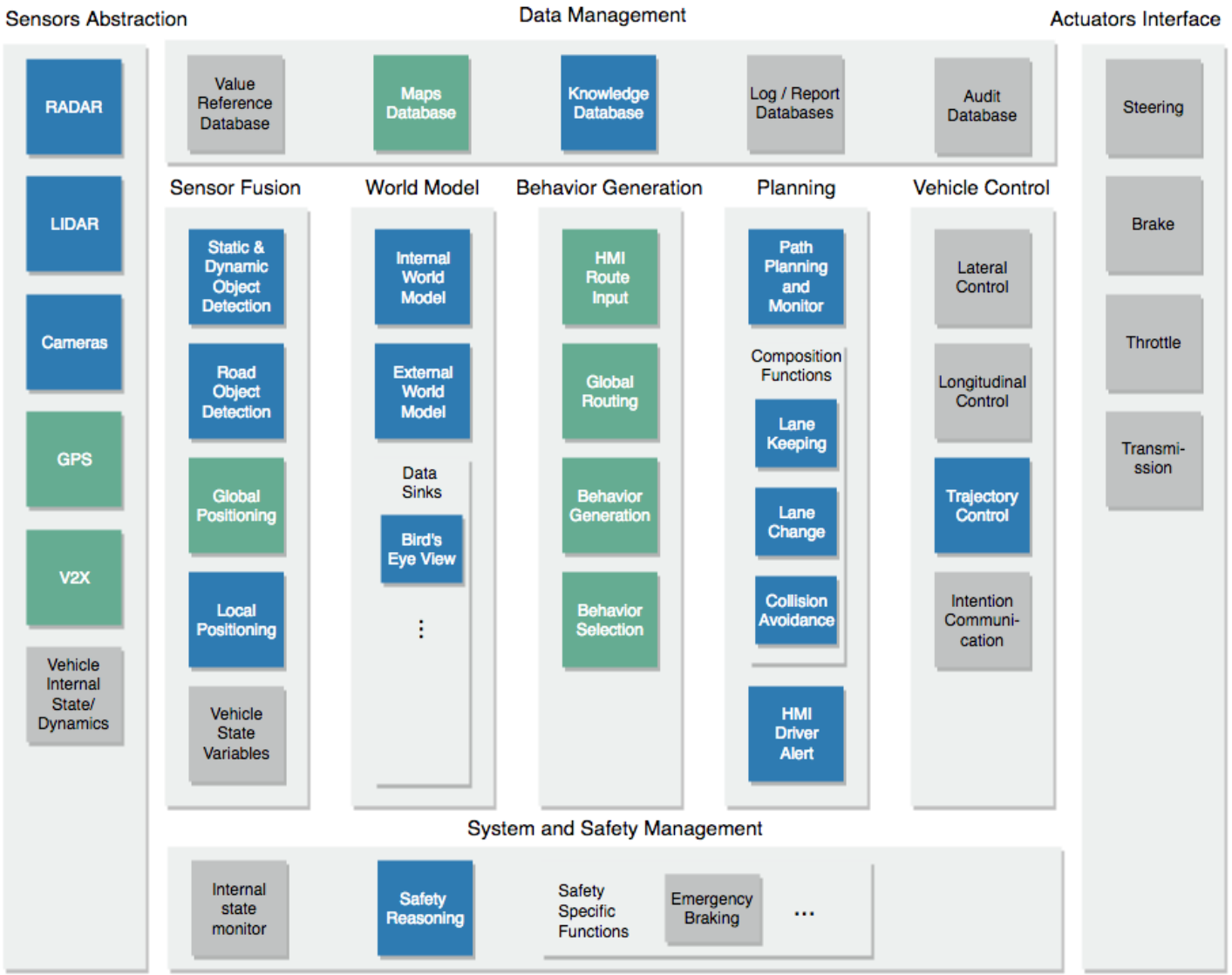
One level higher - strategic decisions



+Data


Sensors Abstraction
Data Management
Actuators Interface
RADAR
LIDAR
Cameras
GPS
V2X
Vehicle Internal State/ Dynamics
Value Reference Database
Maps Database
Knowledge Database
Log / Report Databases
Audit Database
Steering
Brake
Throttle
Transmi-ssion
Sensor Fusion
World Model
Behavior Generation
Planning
Vehicle Control
Static & Dynamic Object Detection
Road Object Detection
Global Positioning
Local Positioning
Vehicle State Variables
Internal World Model
External World Model
Data Sinks
Bird's Eye View
HMI Route Input
Global Routing
Behavior Generation
Behavior Selection
Path Planning and Monitor
Composition Functions
Lane Keeping
Lane Change
Collision Avoidance
HMI Driver Alert
Lateral Control
Longitudinal Control
Trajectory Control
Intention Communi-cation

+Safety


Sensors Abstraction
Data Management
Actuators Interface
RADAR
LIDAR
Cameras
GPS
V2X
Vehicle Internal State/ Dynamics
Value Reference Database
Maps Database
Knowledge Database
Log / Report Databases
Audit Database
Steering
Brake
Throttle
Transmi-ssion
Sensor Fusion
World Model
Behavior Generation
Planning
Vehicle Control
Static & Dynamic Object Detection
Road Object Detection
Global Positioning
Local Positioning
Vehicle State Variables
Internal World Model
External World Model
Data Sinks
Bird's Eye View
HMI Route Input
Global Routing
Behavior Generation
Behavior Selection
Path Planning and Monitor
Composition Functions
Lane Keeping
Lane Change
Collision Avoidance
HMI Driver Alert
Lateral Control
Longitudinal Control
Trajectory Control
Intention Communi-cation
System and Safety Management
Internal state monitor
Safety Reasoning
Safety Specific Functions
Emergency Braking
...

Component Interaction - road to patterns

Component Interaction - road to patterns

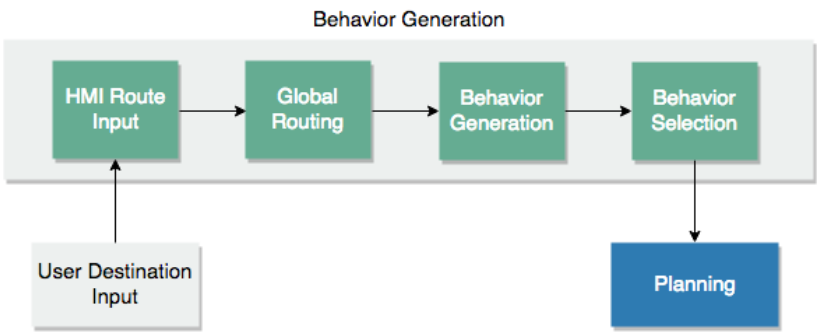


Figure: Component interaction at class level. Pipes-and-filters pattern.

Component Interaction - road to patterns

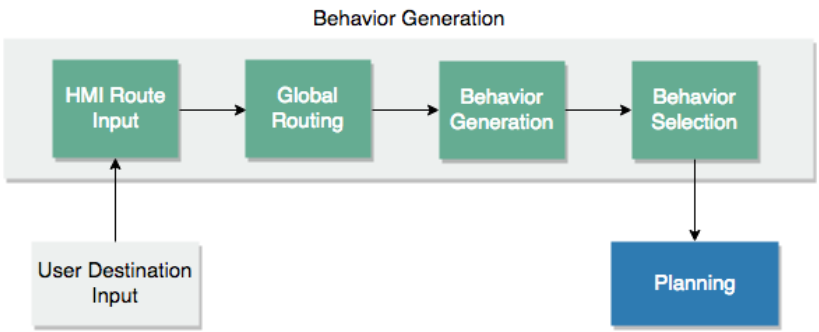


Figure: Component interaction at class level. Pipes-and-filters pattern.

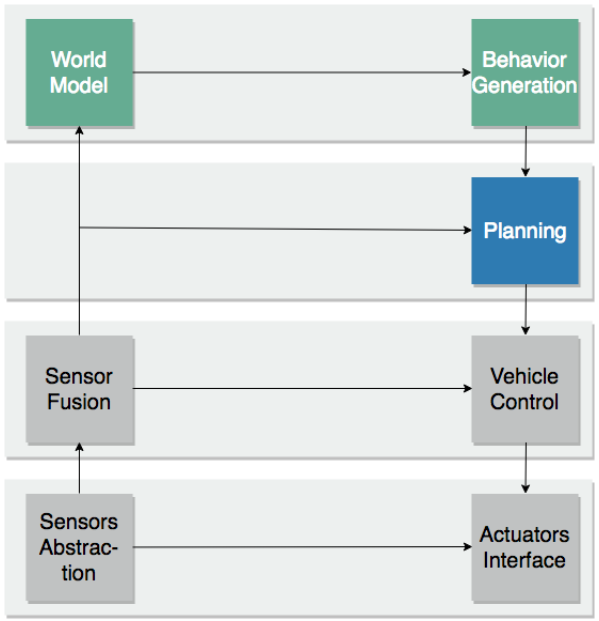


Figure: Hierarchical control. Tee-and-join pipelines pattern.

What next?

- Data about interaction patterns
- Experiments with centralized/decentralized deployment architectures
- (Standardized) component interfaces

References I


Sagar Behere and Martin Törngren. "A functional reference architecture for autonomous driving". In: *Information and Software Technology* 73 (2016), pp. 136–150.


Daniel Kahneman. *Thinking, fast and slow*. Macmillan, 2011.