
# Software Systems Architecture from an automotive perspective

## Experiences from Volvo Cars

Anders Alminger, Vehicle Software & Electronics @ Volvo Cars, Security Class: Proprietary

2020-02-24

# Me


## Anders Alminger

Vehicle SW & Electronics
Volvo Car Corporation

anders.alminger@volvocars.com


© Anders Alminger – “Self portrait”, water colour

# Volvo Cars in figures

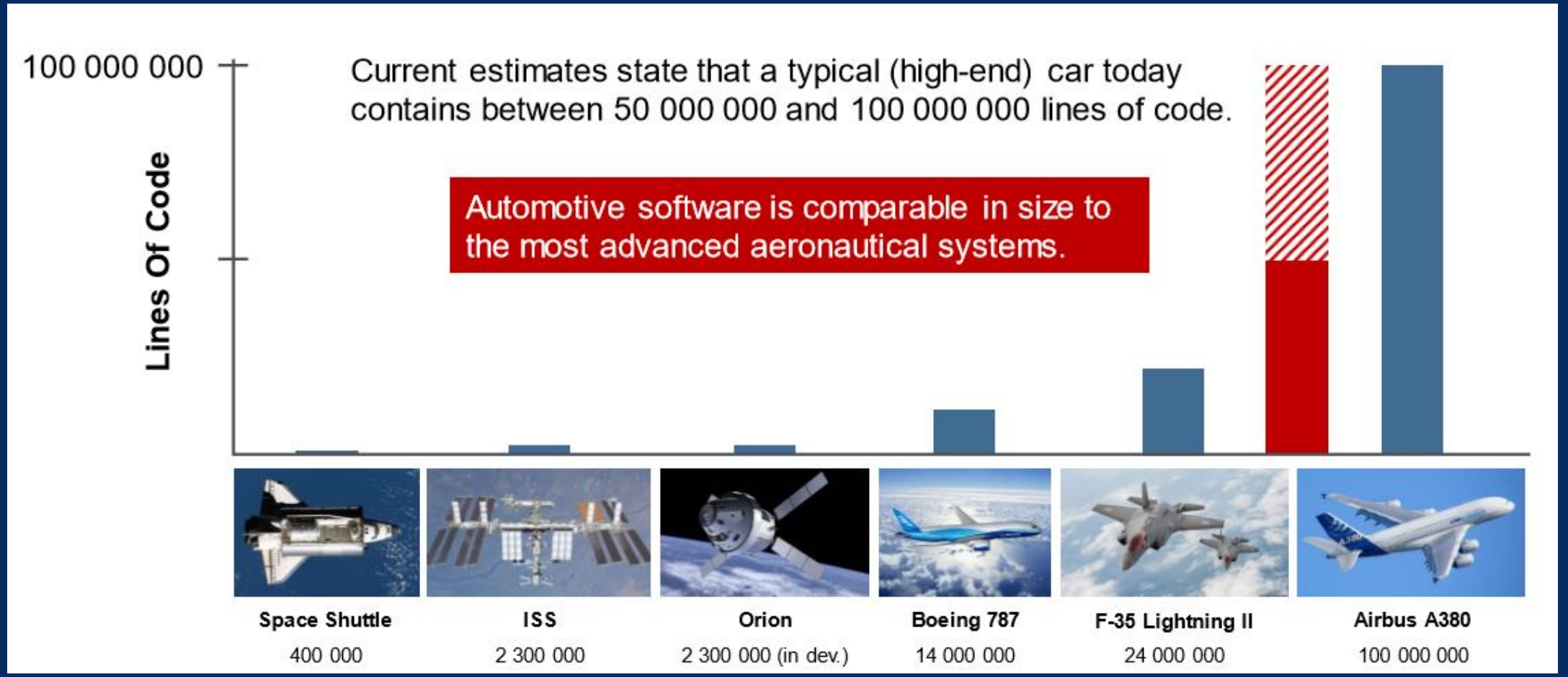
| 705K | 41.5K | 14.3 |
| --- | --- | --- |
| Cars sold in 2019 | Average number of employees | 2019 operating profit (bnSEK) |

# Volvo Cars in short

- Volvo Car Group (Volvo Cars) is owned by Zhejiang Geely Holding (Geely Holding) of China.
- Our group structure comprises Volvo Cars and our related direct consumer businesses: car subscription service Care by Volvo and mobility company M.
- Volvo Car Group also includes the sizeable stakes in our strategic affiliates:
  - electric performance brand Polestar (50% owned by Volvo Cars),
  - new Chinese car brand LYNK & CO (30%)
  - and software company Zenuity (50%).

# Volvo cars global presence

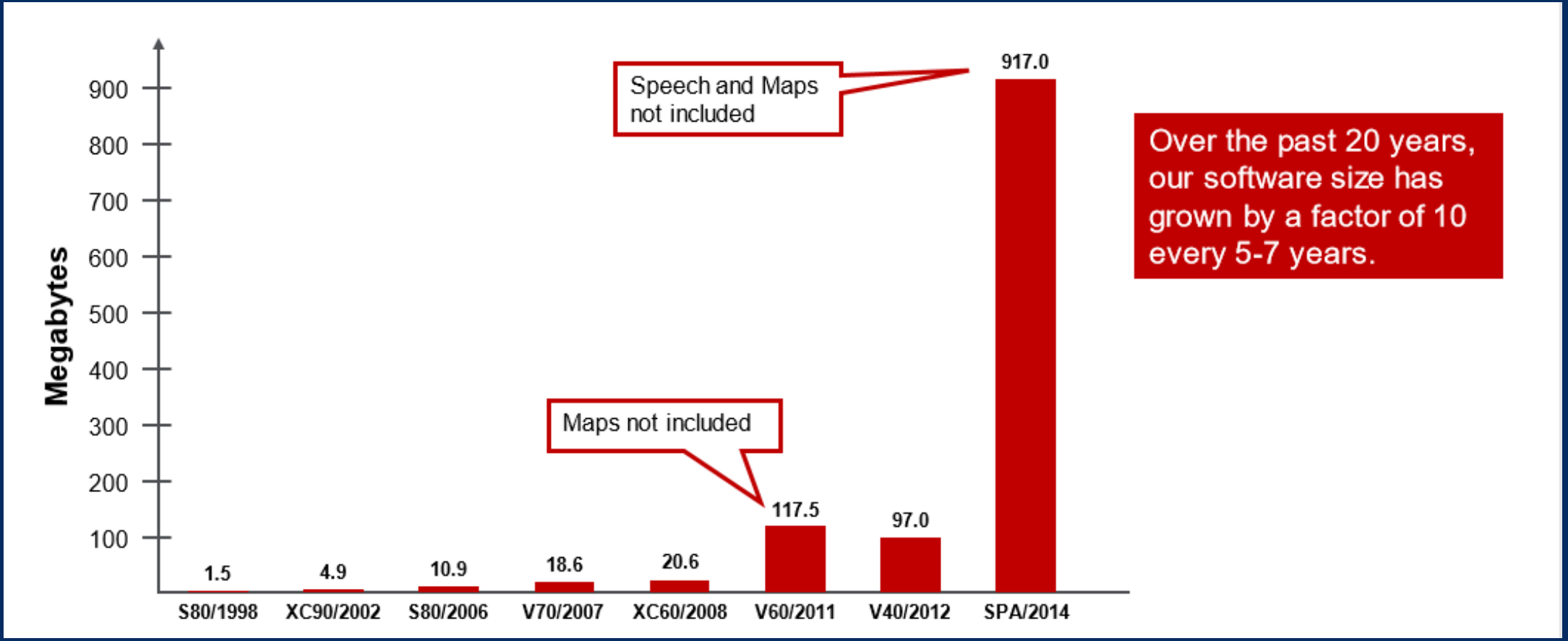
# Software intensive systems




Reference: Martin Hiller (Volvo Cars), ICSA 2017 KEYNOTE – Surviving in an increasingly computerized and software driven automotive industry

# Software* size evolution at Volvo Cars

*) Downloadable Software



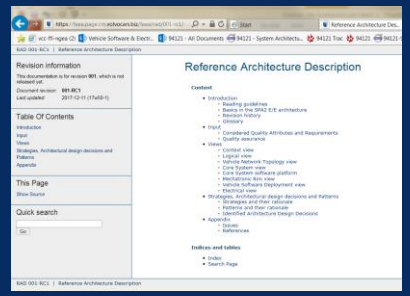

Reference: Martin Hiller (Volvo Cars), ICSA 2017 KEYNOTE – Surviving in an increasingly computerized and software driven automotive industry

Terms Used

- Architecting

- Architecture

- Architecture Description



# What do we mean when we say Architecture?

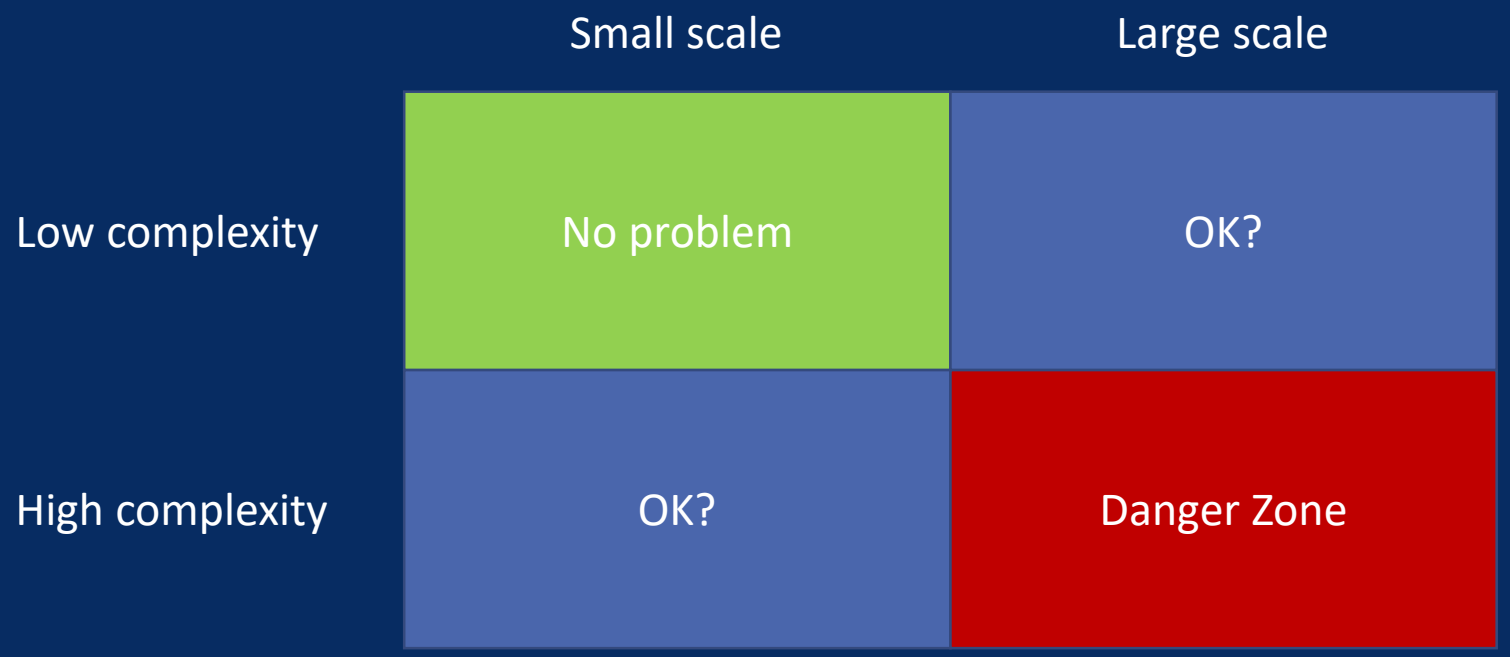
- ”Every system has an architecture, intended or not”
  [Dennis Selin, KnowIT]
- “<system> fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution”
  [IEEE 42010:2011 Systems and software engineering – Architecture Description]
- “The software architecture of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them and properties of both.”
- “The architecture is a bridge between (often abstract) business goals and the final (concrete) system”
  [Bass, Clements, Kazman, Software Architecture in Practice, 3rd Edition]


*No, it isn't. And the Architecture Description is not the Architecture.*

**architecture description AD =**
**=** work product used to express an architecture

**architecture framework =** conventions, principles and practices for the description of architectures established within a specific domain of application and/or community of stakeholders

# Why Architecture?



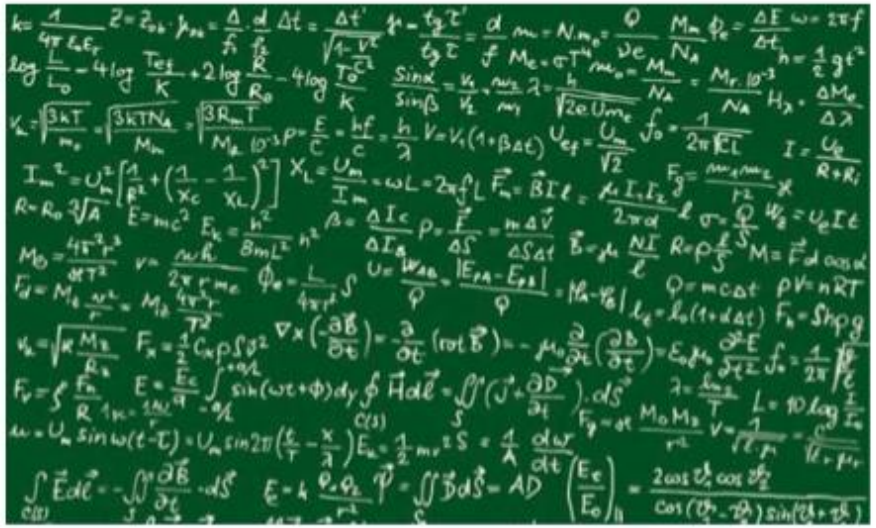

Reference: Ulrik Ekedahl, Aikane Tech

# Why Architecture (2)

- If you want to control the properties of the system
- If you want to manage complexity
- If you need to reuse components
- If you need a way to reason about the system on another abstraction level then code

# About complexity

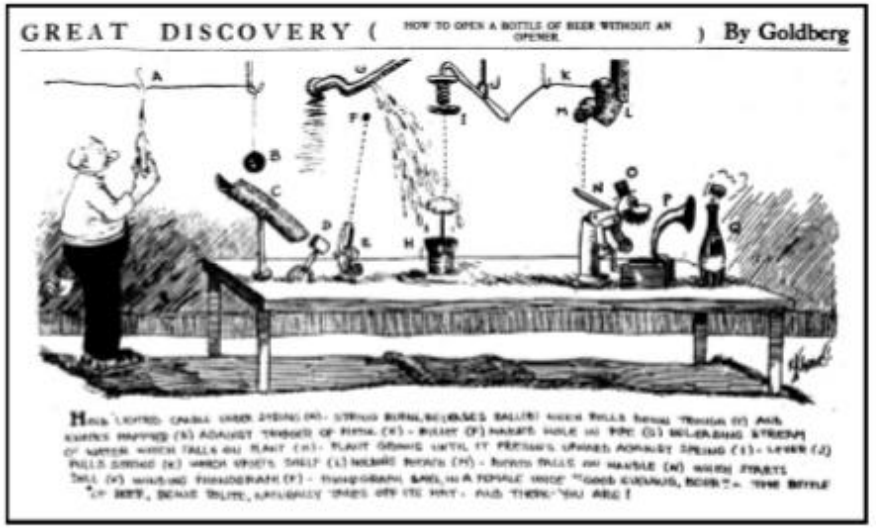
## Essential



The inherent difficulty of the problem to be solved. Some things are just harder than others...

VS

## Accidental




The solution to a problem may sometimes be more complex than it needs to be...

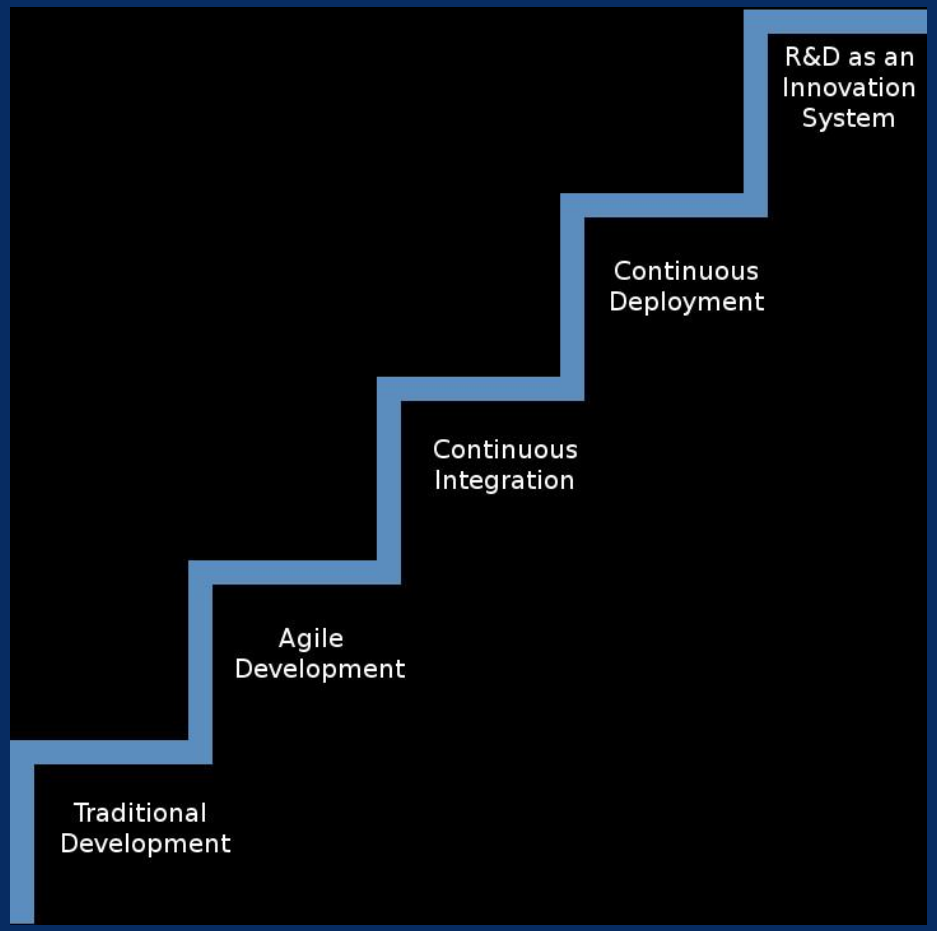
Some sources of accidental complexity:
Carry-over & technical debt...
Project constraints (time & budget)...
Lack of skills & experience...
Organisational structure (Conway's Law)...

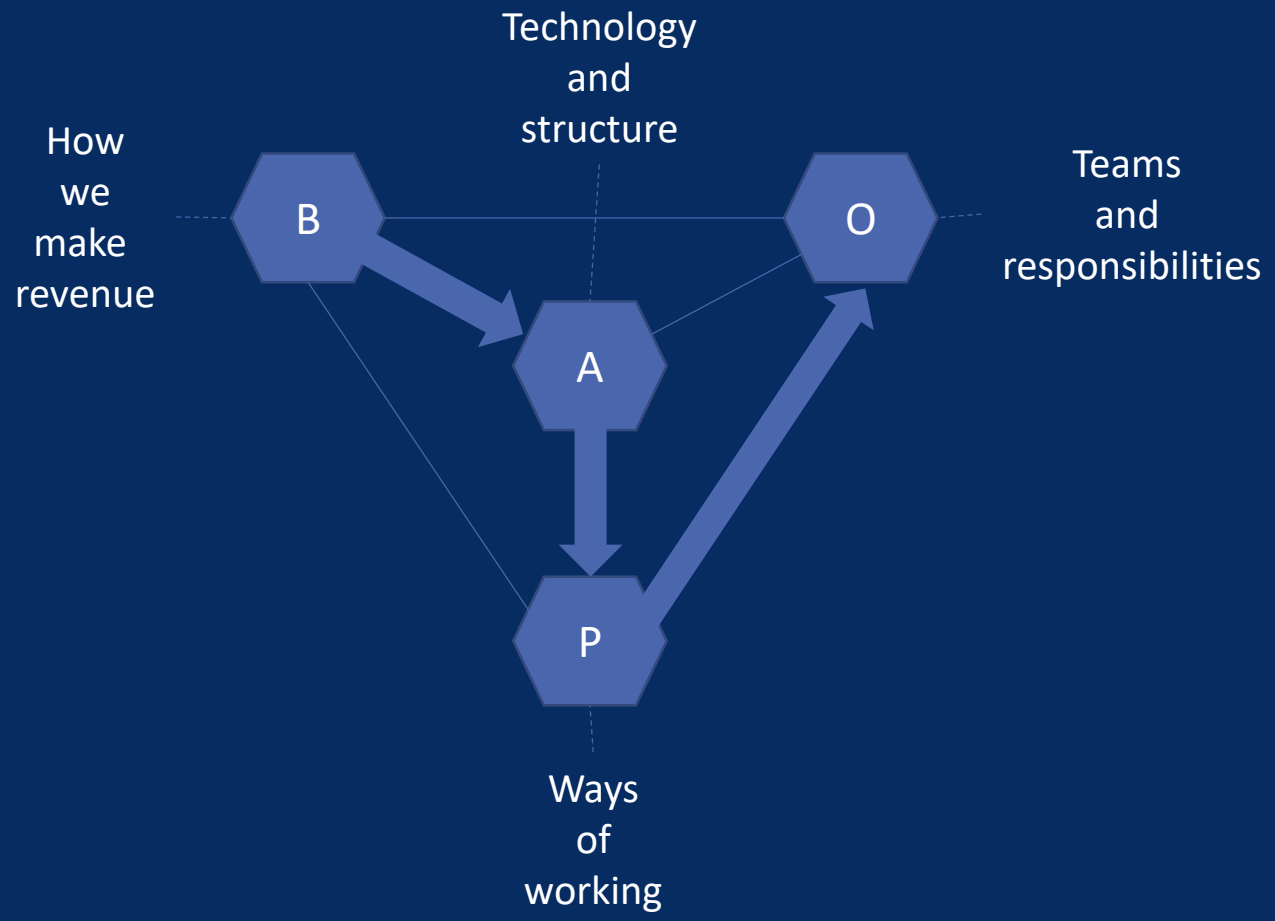
Architecting

"Jazz music is one of the few human activities that succeeds in combining responsibility for the **collective** with **individual** freedom. “

Jazz pianist and composer Lars Jansson

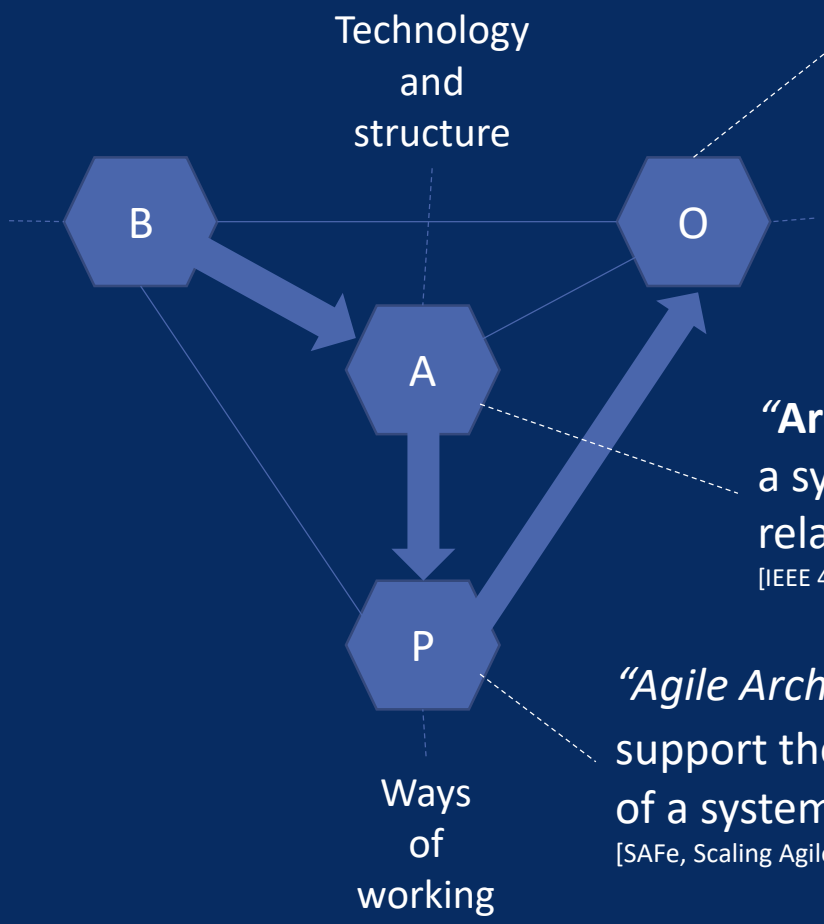
# Why Agile & how is Architecture affected?




References:
F. van der Linden, J. Bosch, E. Kamsties, K. Kansala, and J. H. Obbink, “Software product family evaluation,” in SPLC, 2004, pp. 110–129.
Helena Holmström Olsson, Hiva Alahyari and Jan Bosch, ”Climbing the “Stairway to Heaven”, 2012 38th Euromicro Conference on Software Engineering and Advanced Applications

# Why Agile & how is Architecture affected?



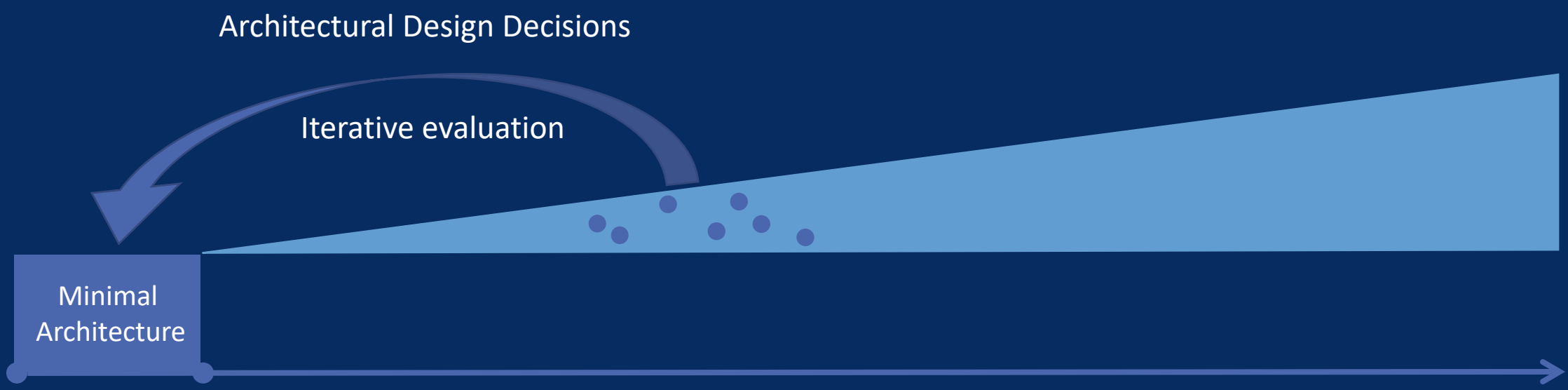

The organizational structure and distribution of team responsibilities are guided by the architecture

Teams and responsibilities

"**Architecture** <system> fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution ."
[IEEE 420101:2011]

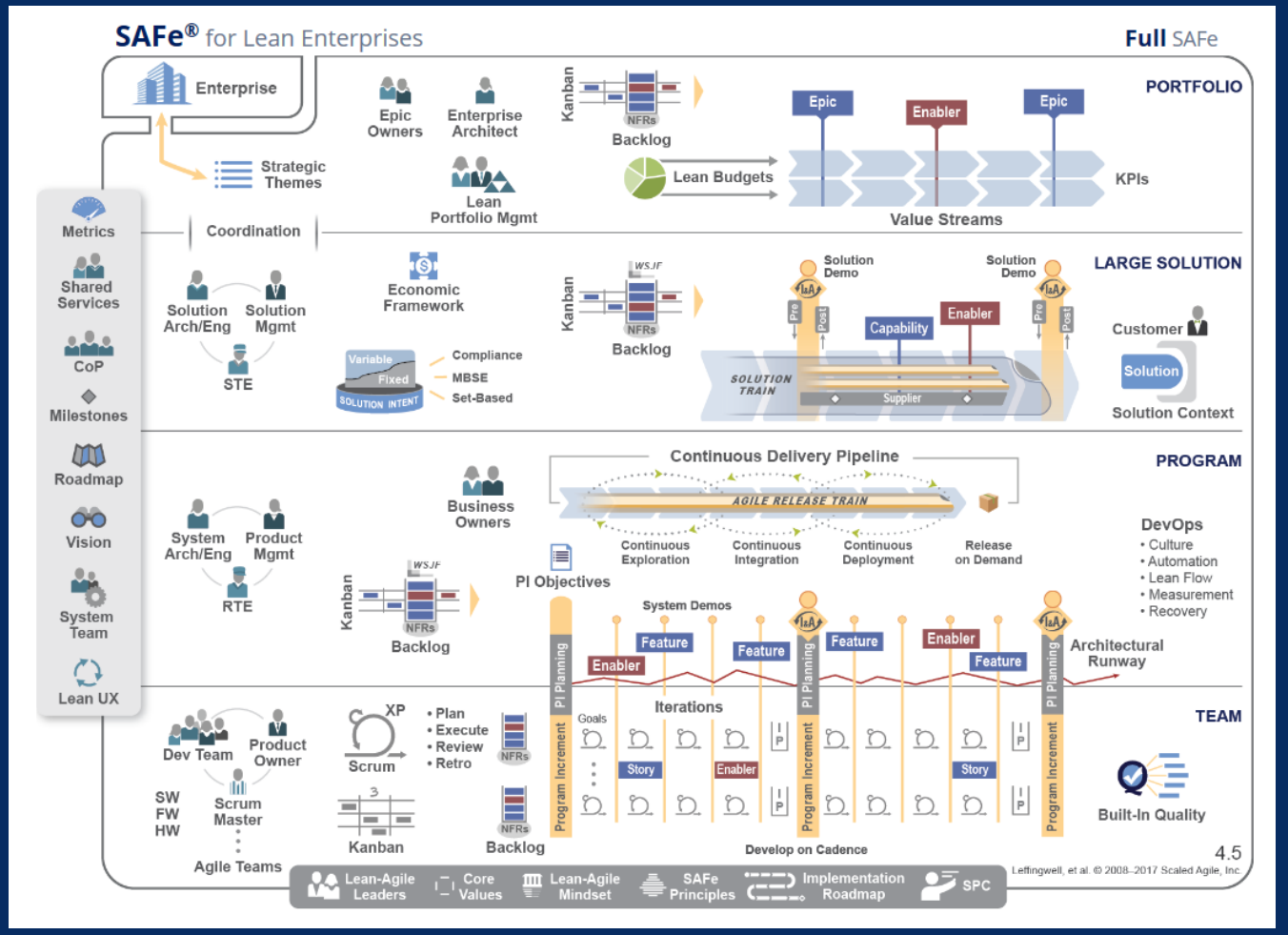
*"Agile Architecture* is **a set of values and practices** that support the active evolution of the **design** and **architecture** of a system while implementing new system capabilities."
[SAFe, Scaling Agile Framework]

# Agile Architecting




= Intentional Architecture
”Small Design Up Front”

= Emergent Design
”just-in-time architecture”

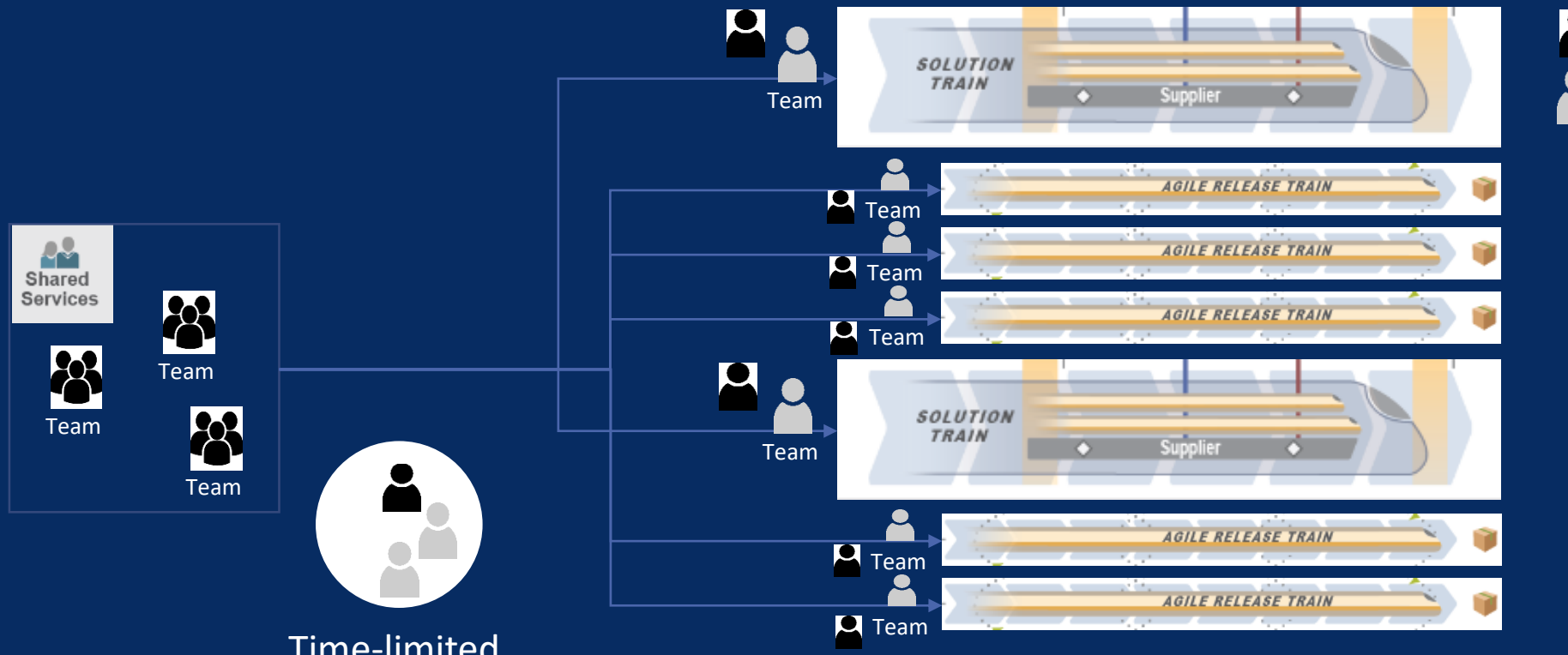



# Full SAFe as a base to scale

- ~ 6000 people in P&Q (=”R&D”)
- 11 large solutions (product streams)
- ~ 50 Agile Release Trains
- ~ 500 Agile Teams

Reference:
Darko Durisic, “Shifting from Traditional Development to Full Agile in a Large-Scale Mechatronics Company”, Agile Transformation, Berlin, 2018

# Teaming up the architecture community

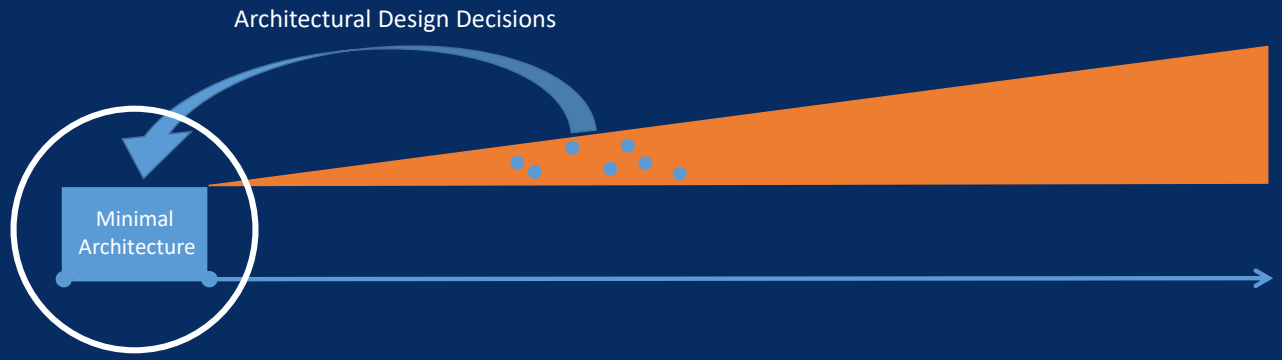


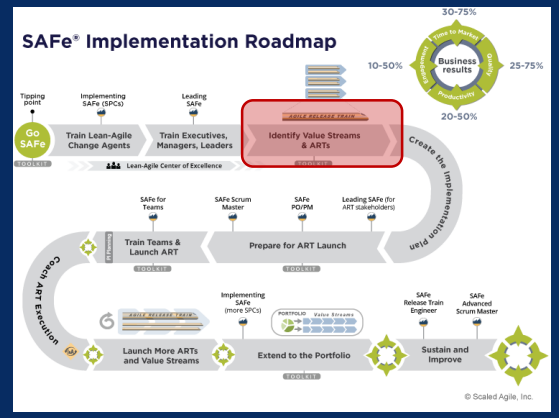

Time-limited cross-train teams

Etc. for 11 Solution Trains ...

Reference:
Darko Durisic, “Shifting from Traditional Development to Full Agile in a Large-Scale Mechatronics Company”, Agile Transformation, Berlin, 2018

# The Minimal Architecture

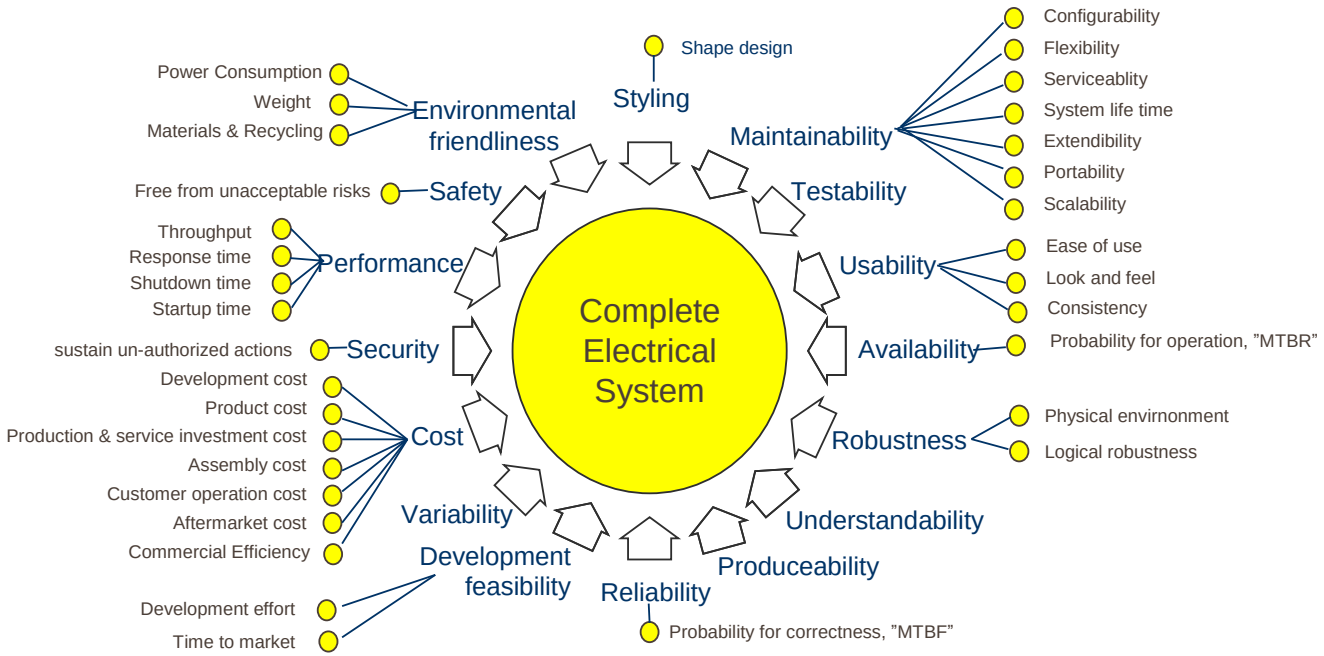



- To limit the architecture we base it on tactics [1] based on the quality attributes derived from business goals
  - To promote **development speed** and **quality**, going agile could be described as an architecture tactic resulting in the Solution/ART structure
    - Risk: Old communication paths are cemented by copying the current line organization into the Solution/ART structure [2]
  - To promote **complexity management** and **variability**, modularization might be another architecture tactic of choice

References:
[1] Rick Kazman, Michael Gagliardi, William Wood: Scaling up software analysis, The Journal of Systems and Software, 2011
[2] Melvin Conway, 1967: Conway's law

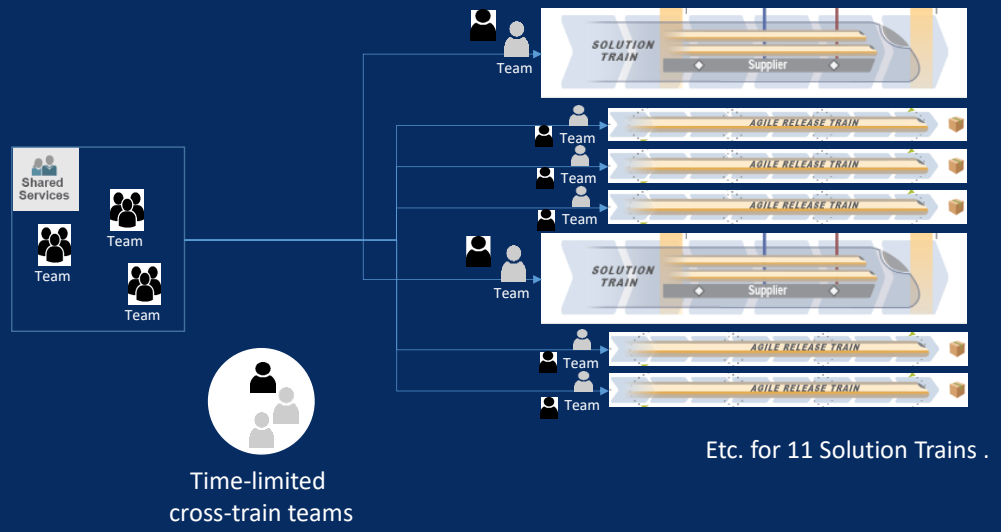
# Quality Attributes



What about the Architect role?

# Two types of architects [1]

- Maker and keeper of big decisions
- Mentor, troubleshooter and prototyper

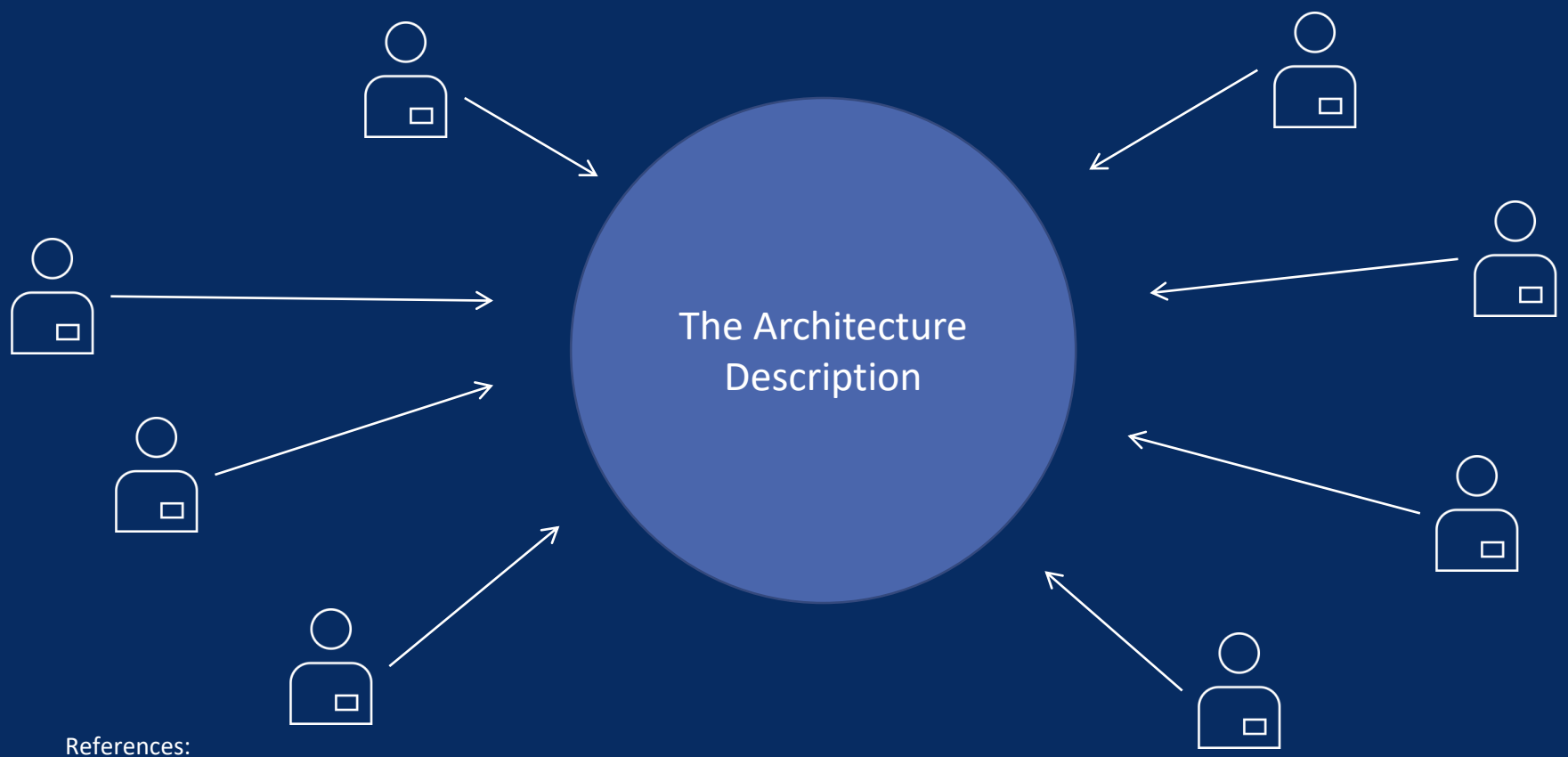



# Enabling skills for successful architects

- Good collaboration – systems thinking
- Systematically work on the distribution of architecture skills [2]
- Take architectural design decisions where the knowledge is – architectural reasoning beats organizational position
- Mutual mentorship – learn from each other
- Establish an arena for architectural reasoning – e.g. the Architecture Description

References:
[1] Martin Fowler, 2004
[2] George Fairbanks, Just Enough Software Architecture, A Risk-Driven Approach, 2010

# The Architecture Description

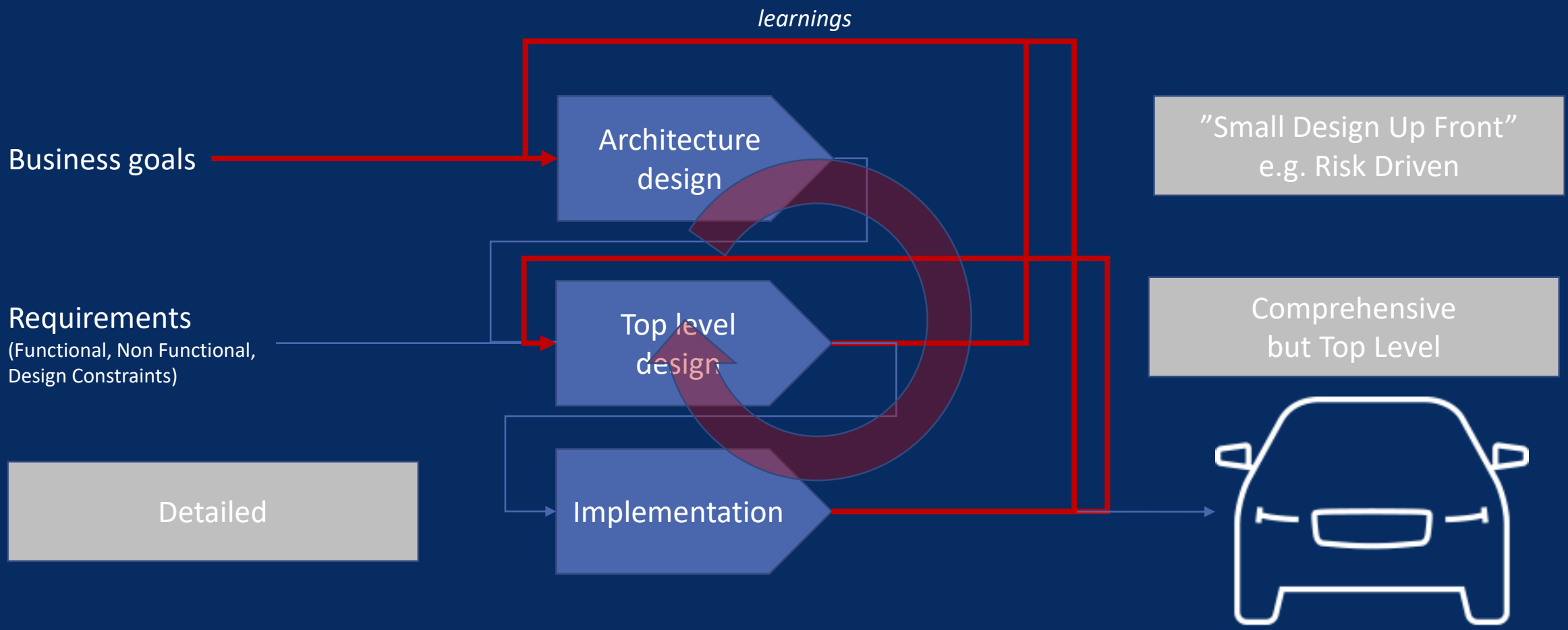



The Architecture Descriptions is a boundary object [1]

”Boundary objects are objects which are both plastic enough to adapt to local needs and constraints of the several parties employing them, yet robust enough to maintain a common identity across sites” [2]

References:
[1] Rebekka Wohlrab et al., 2018
[2] Susan Leigh Star and James R. Griesemer. 1989

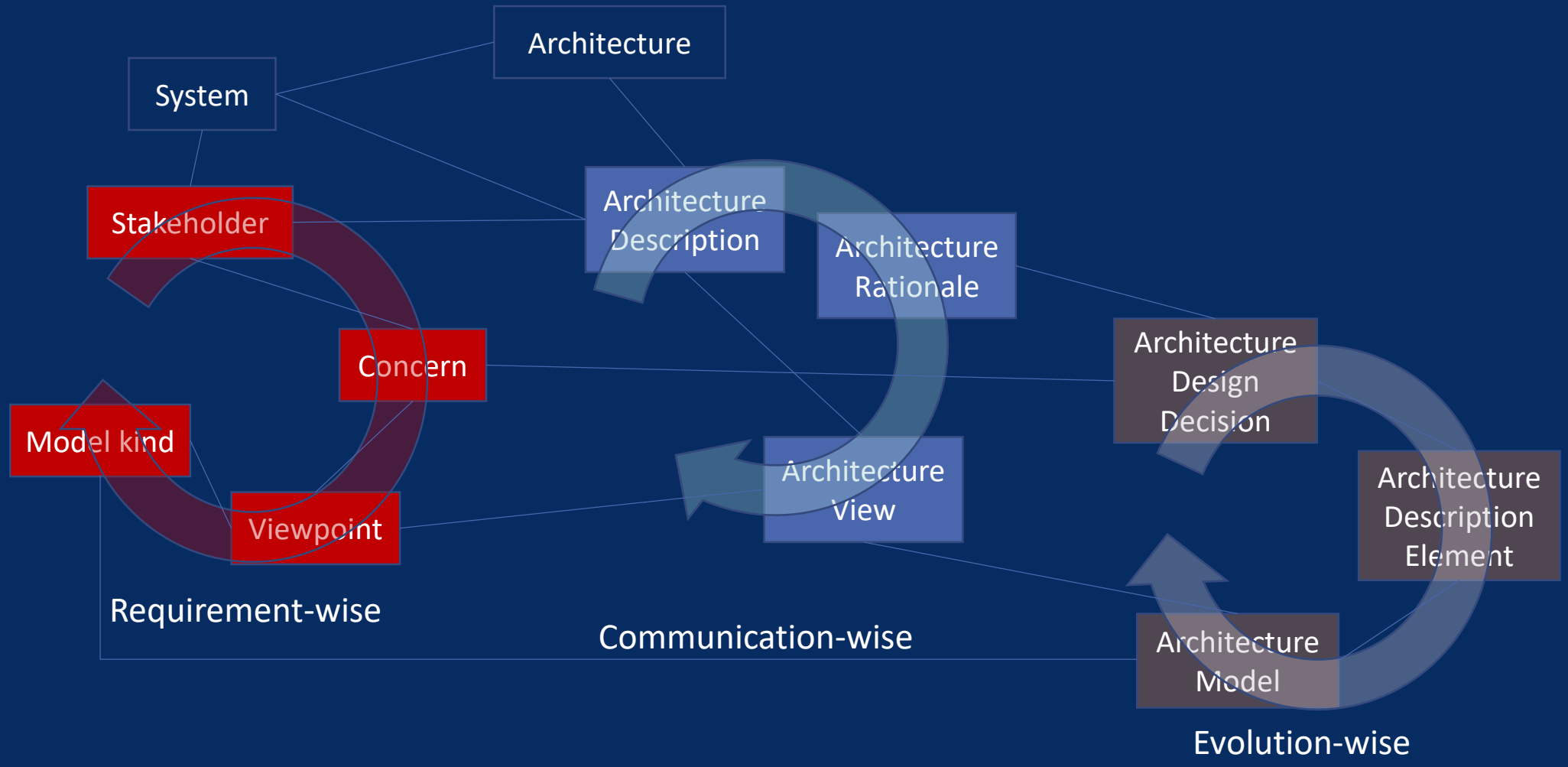
# Design Mindset



"**Agile Engineering** is a human activity that succeeds in combining responsibility for the collective with individual freedom. "

Paraphrasing jazz pianist and composer Lars Jansson


Ceci n'est pas une pipe.


Architecture
System
Stakeholder
Architecture Description
Architecture Rationale
Concern
Model kind
Viewpoint
Architecture View
Architecture Design Decision
Architecture Description Element
Architecture Model
Requirement-wise
Communication-wise
Evolution-wise

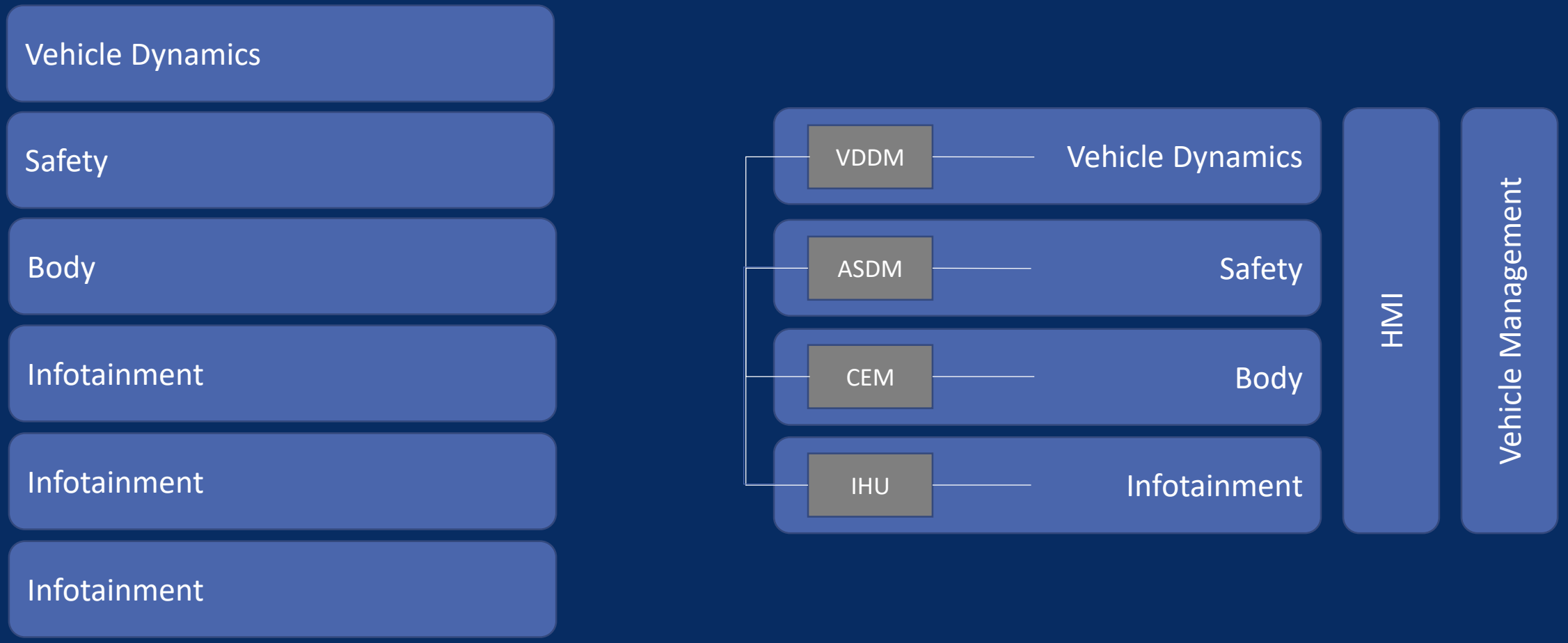
# Typical Stakeholders and their concerns

- Customers
- End-users
- Developers
- Maintainers
- Management

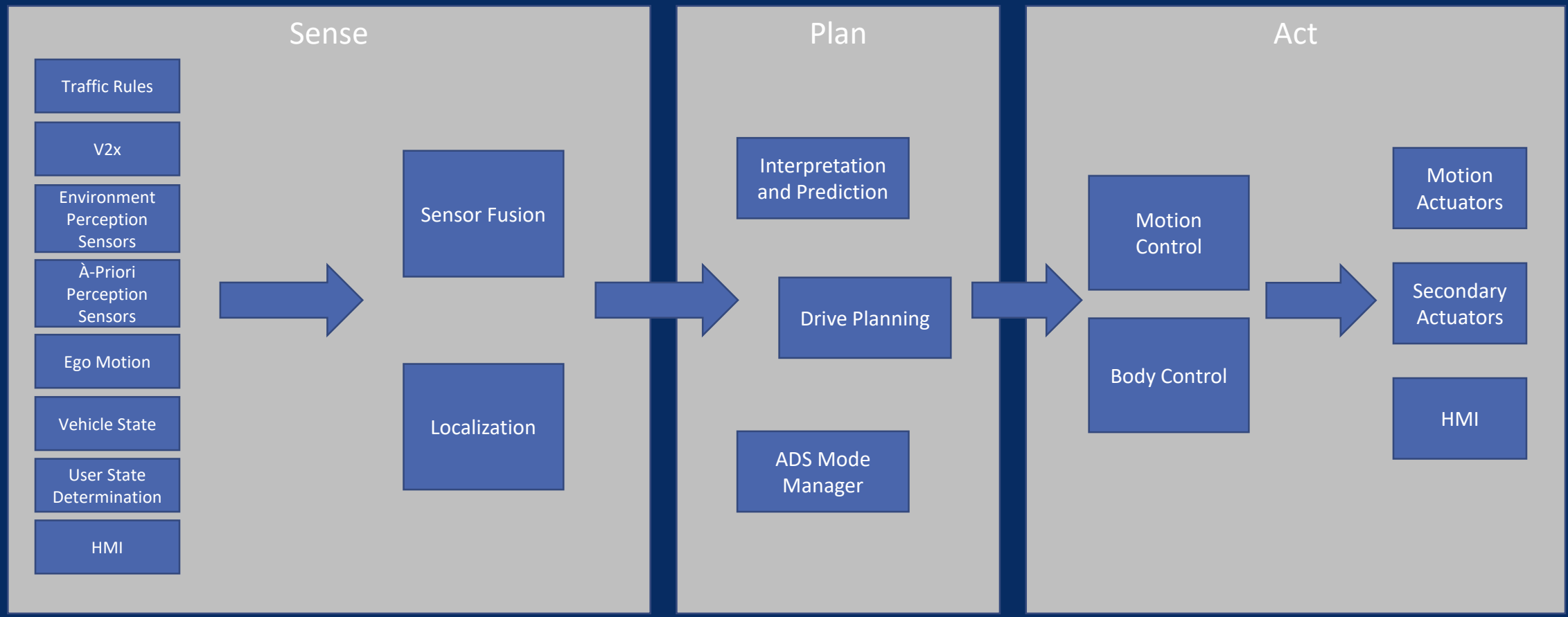
- *As a consumer I want to update my vehicle with new features in the same pace as they are launched*
- *As a user I want the vehicle to be safe and dependable*
- *As a developer I want to be able to reason about the system and avoid building the wrong system*
- *As a maintainer I want to be able to fault trace the system*
- *As a manager I want the development community to build a system with quality meeting quick changes in the market*

- i.e. we need viewpoints that addresses prioritized stakeholder concerns
  - Remember: we want the architecture minimal, not comprehensive, i.e. some stakeholder concerns are dealt with in Top Level Design and Implementation (the terms can be generalized outside architectural scope)

# Logical & physical domains



# Unreal Example from Autonomous Cars



# A variant of Pipes & Filters Pattern


What does my reality look like?

What should I do to get my reality as I want it to be?

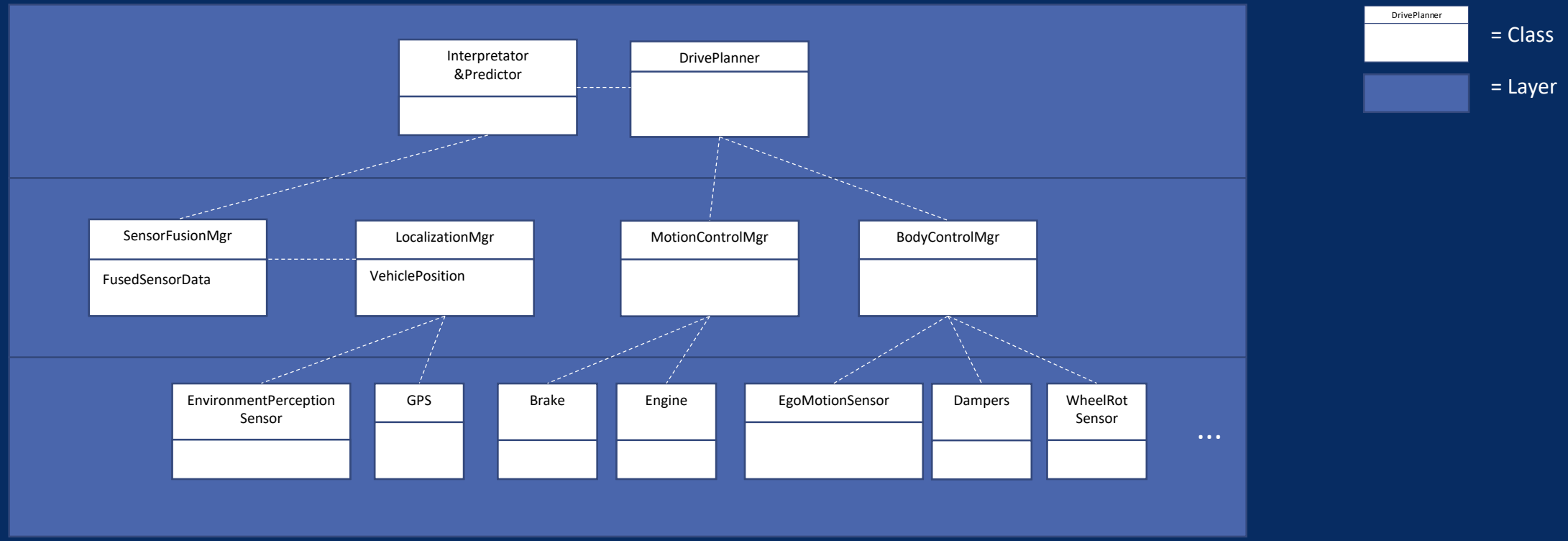
Let’s do it!

**Sensors**
provide us with or world view.

**Algorithms**
for analysis and control.

**Actuators**
allow us to affect our surroundings.

# Logical View – Layer Pattern (Hardware Abstraction)

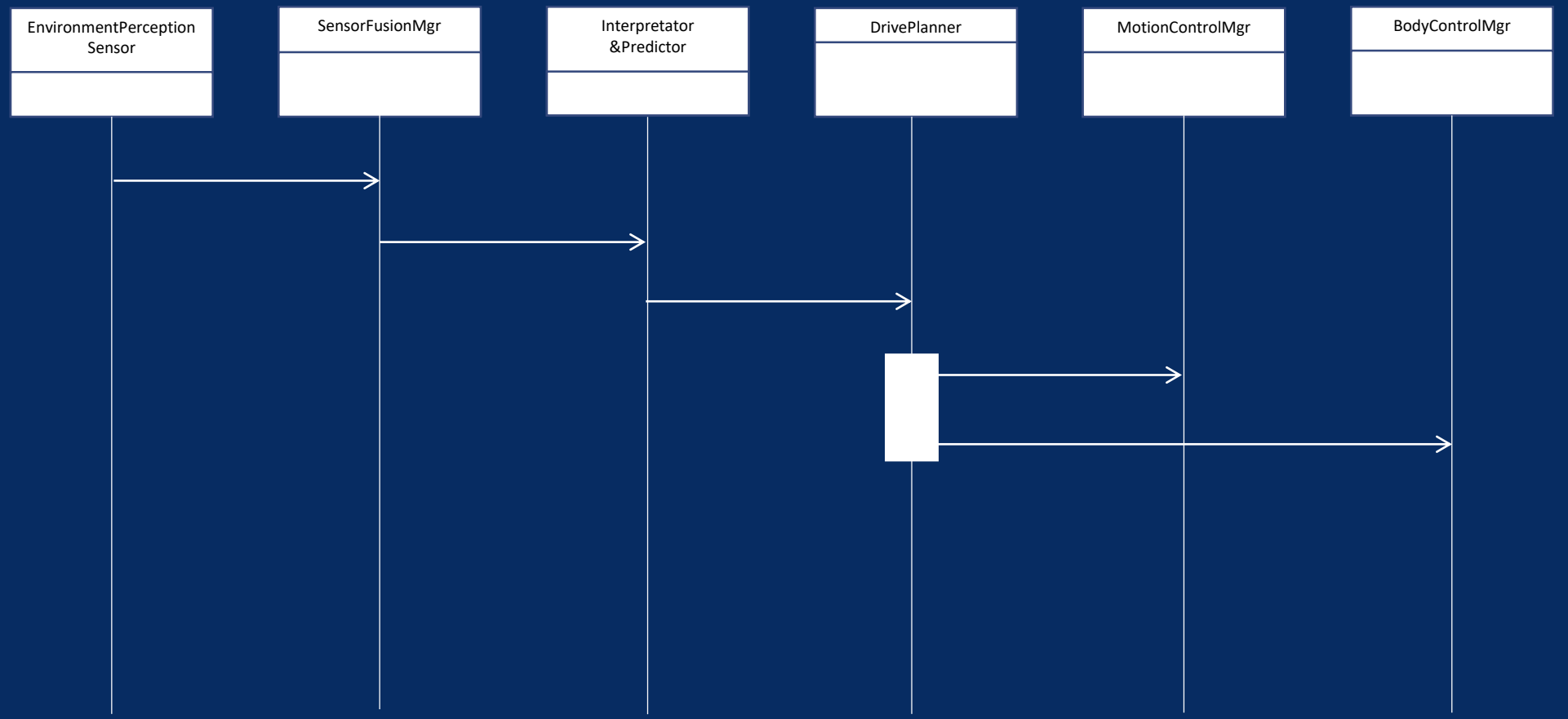



Reuse

Standardisation

Dependencies are kept local

# Using Sequence Diagrams to show the typical interaction logic between classes in the system



# Architecture Decisions
# - how it might look

1. Centralize computational power

   Rationale: to enable faster feature deployment

2. Centralize management of application in domain managers

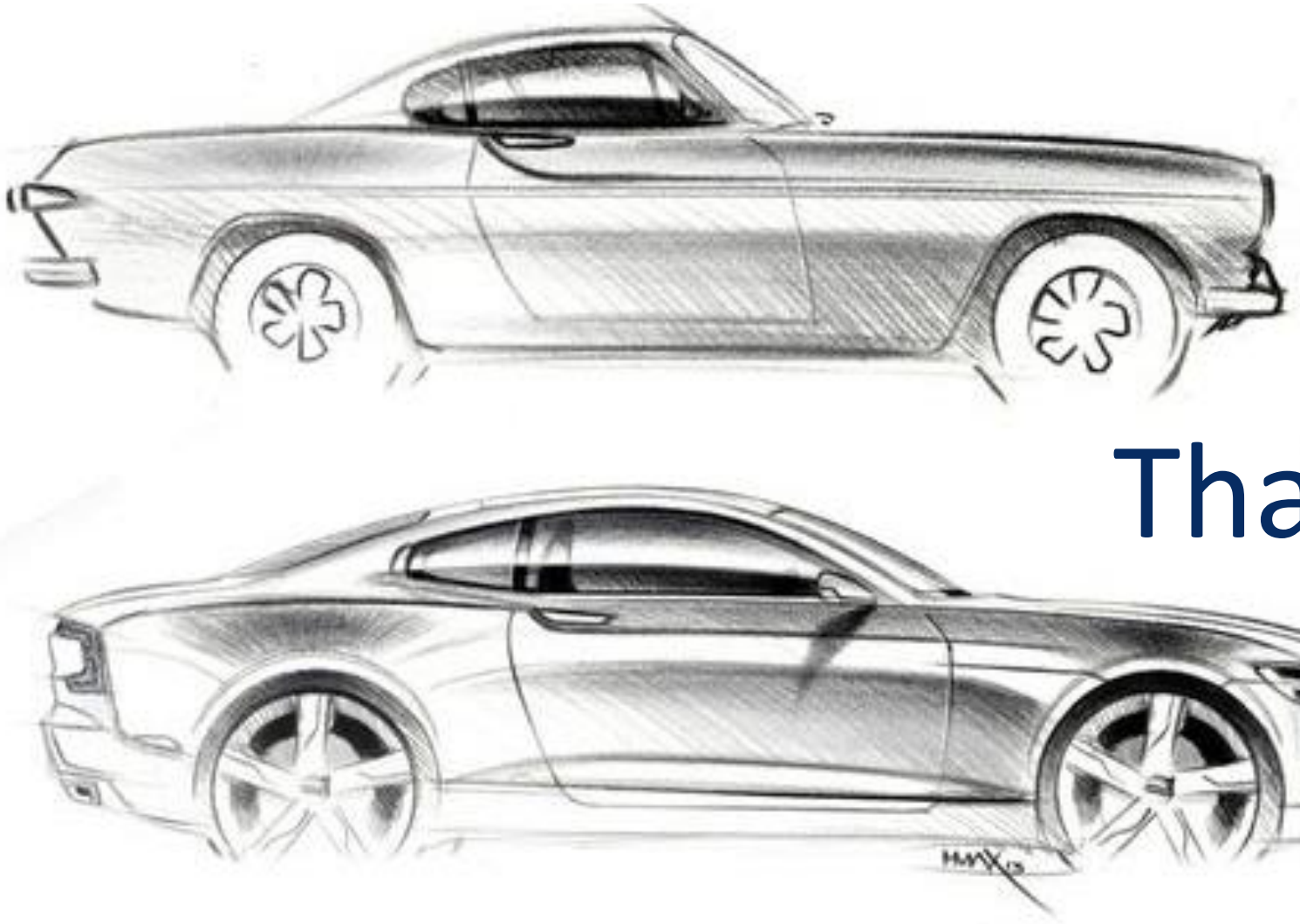
   Rationale: Dependencies are kept local. Accidental complextity is limited.

3. 3 layers with Hardware Abstraction

   Rationale: to decouple hardware from software and to decouple software with different life cycles

4. FlexRay (a deterministic communication protocol) for backbone communication

   Rationale: to enable predictive communication behavior and high performance



# Thank you!