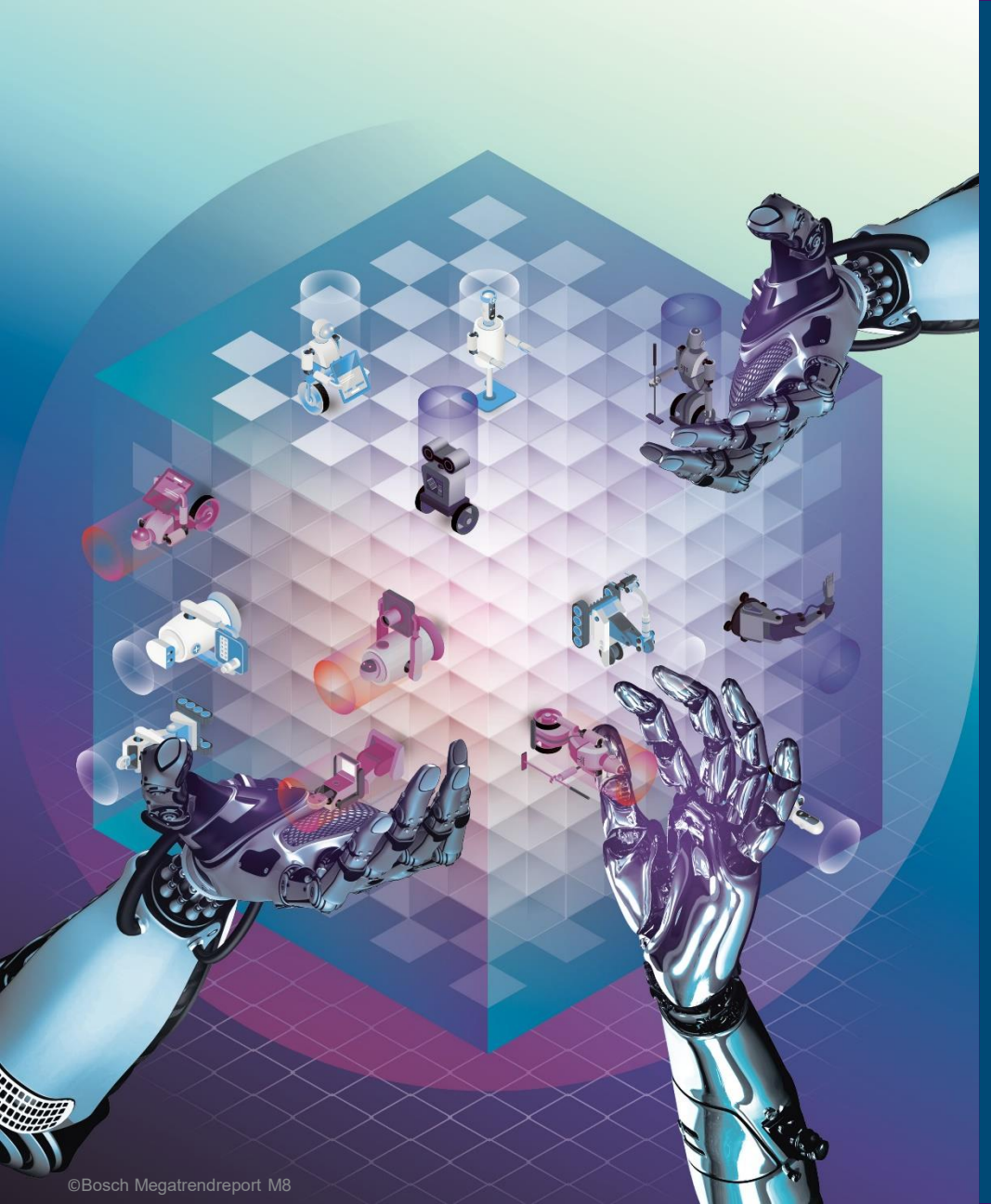



ACM SIGSOFT Summer School
for Software Engineering in Robotics 2025

# Software Engineering for Reliable Autonomous Robots:

## Approaches & Challenges from an Industrial Perspective

Dr. Michaela Klauck
Bosch Research

BOSCH

# Who I am
## Motivation

**Dr. Michaela Klauck**
michaela.klauck@de.bosch.com

- PhD at Saarland University in Formal Verification of Cyber-Physical Systems
  - Probabilistic model checking
  - Automated planning
  - NN verification
- Research Engineer for Verification of Planning & Decision-Making in Autonomous Systems at Bosch Research
- Principal investigator in EU Horizon project CONVINCE at Bosch Research
  - Robust & reliable deliberation of autonomous robots
- Formal verification of behavior planners of autonomous cars

# Software Engineering for Reliable Autonomous Robots
## Agenda

- Short introduction to Robotics at Bosch
- Quality of SW-intensive Products: Theoretical Background
  - Norms for quality assurance and agile processes
- Learnings and best practices along the typical robot SW stack
  - Microcontrollers
  - Skills / Capabilities
  - Simulation
  - Deliberation / Decision-making
- Example for classical model-checking with Spin in robotics software
- The Why and How of ROS at Bosch

# 01

# Robotics at Bosch

# Who we are
## Our company in figures

**In 2023**

**91.6**

billion euros
sales revenue

**4.8**

billion euros EBIT
from operations

**429 400**

Bosch associates
worldwide at year-end
(approx.)

**470**

subsidiaries and regional
companies (approx.) in
more than **60** countries





BOSCH

# Who we are
## Our business sectors

**Mobility**

**Industrial Technology**

**Energy and Building Technology**

**Consumer Goods**

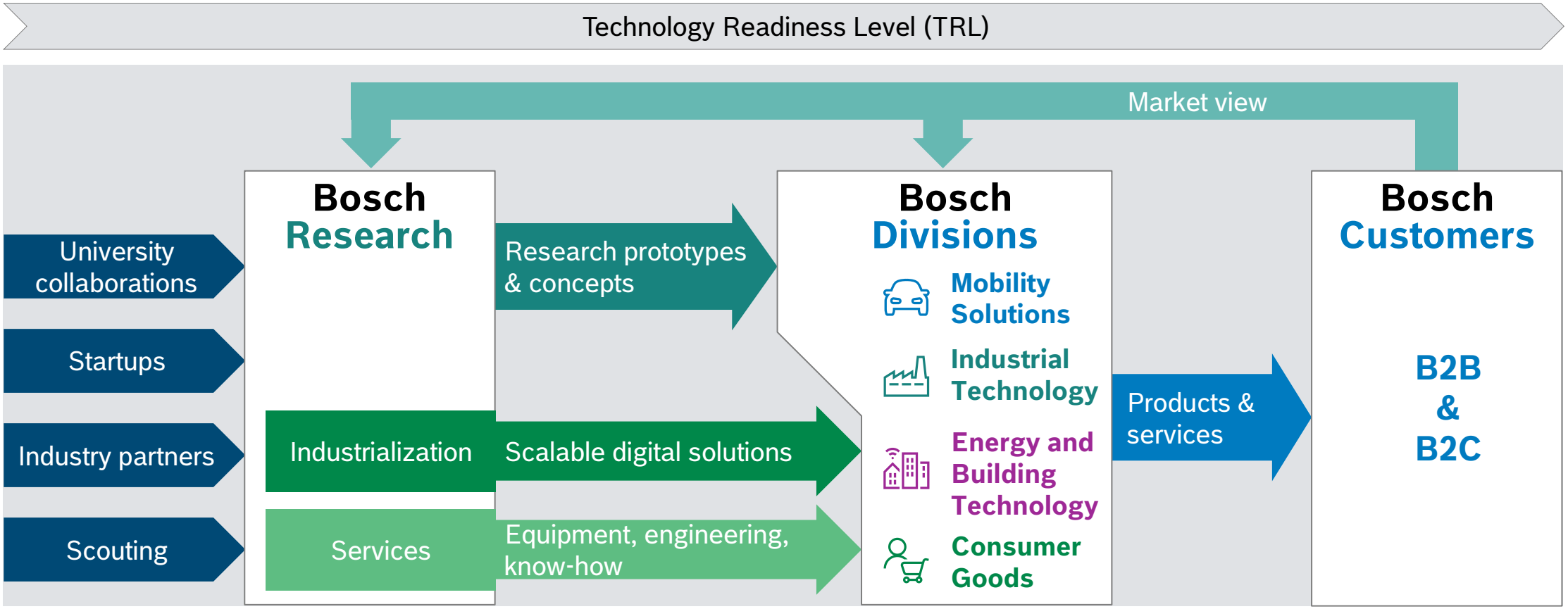




BOSCH

# Innovation @ Bosch

## Role of Bosch Research in the innovation landscape



# Bosch Research

## Organization, figures and locations

**In 2023**

**~1 800**

highly specialized employees including PhDs

**92%**

of employees are scientists

**149**

PhD students

**63**

nations within associates

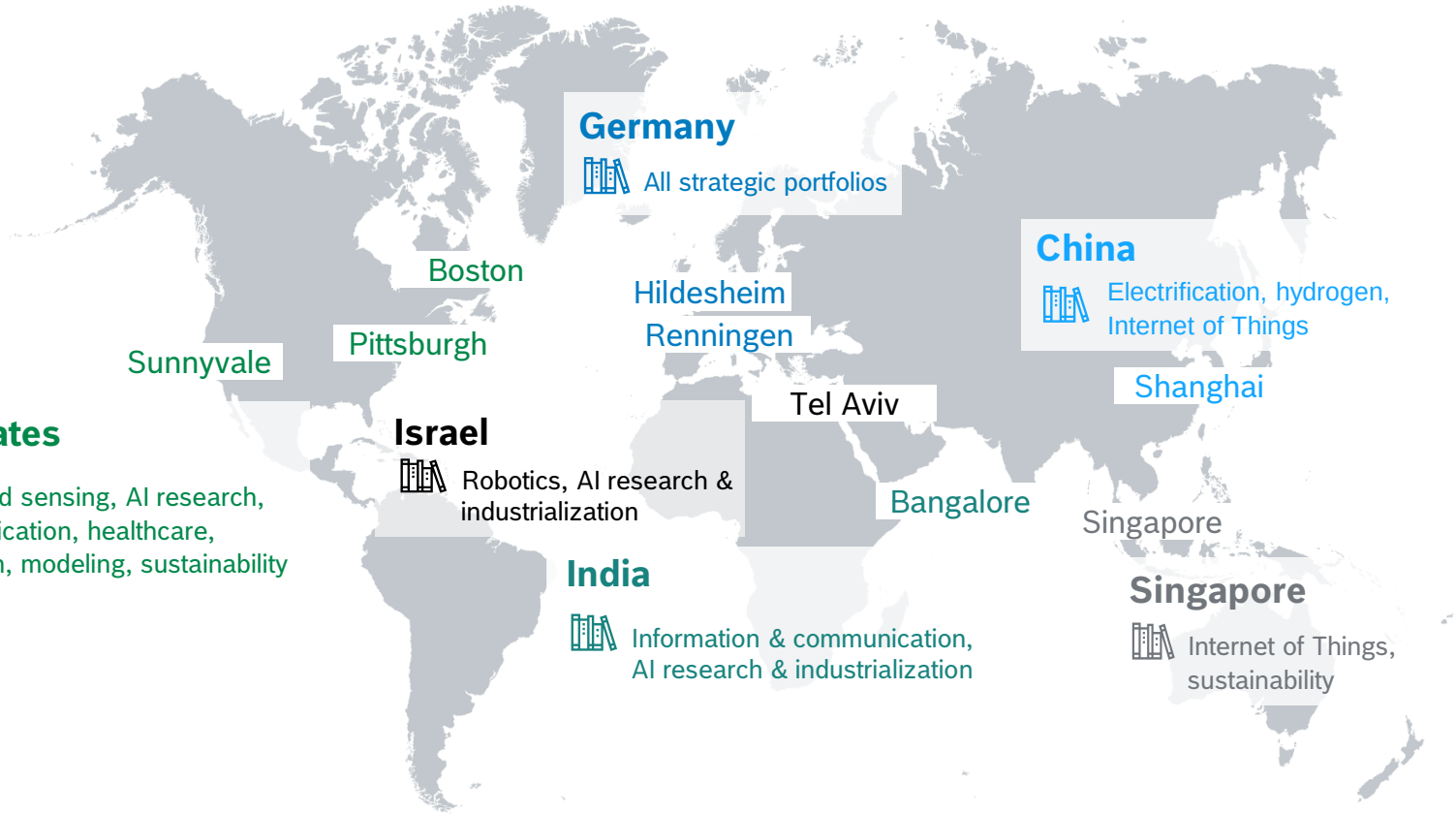
**9**

top research facilities around the globe

**~1 500**

patent filings

# Bosch Research
## Leveraging our international setup




- **Connect to the best**
- **Connect to RB local**
- **Local Tech markets**
- **Regional economics & talents**
- **Decoupling**





BOSCH

# Strategic Research Portfolios & Business Vertical

**Automated Intelligent Driving (AID)**
Sensing, perception, prediction, planning, systems & infrastructure for L2-L4

**Artificial Intelligence Methods (AIM)**
Computer vision, language processing (NLP), AI data loop, AI enabling

**Chemical Energy Conversion (CEC)**
Fuel cells (stationary & mobile), electrolysis, hydrogen storage

**Electrified Mobility and Systems (EMY)**
eDrives, power electronics, x-by-wire, energy mgmt., EV thermal systems

**Healthcare Solutions (HCS)**
Point-of-care lab diagnostics, liquid biopsy, next-gen sequencing

**Information and Communication Technology (ICT)**
SW engineering, distributed systems, security, safety & privacy, HMI

**Applied AI & SW Business (BIS)**
Industrialization and service

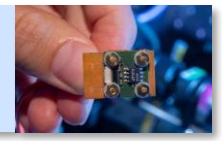
**Internet of Things @ Life (IOT)**
Mobility and residential solutions for consumer applications

**Modeling, Simulation, Optimization (MSO)**
Virtual product design, virtual validation and use-phase monitoring

**Production Systems (PRS)**
Production technologies, AI in production, internet of production

**Sustainability (SST)**
Circular economy, climate change mitigation incl. $CO_2$ removal

**Smart Sensors and HW Systems (SSY)**
SoC, embedded AI, HW-/SW-co-design, sensors (MEMS, quantum)

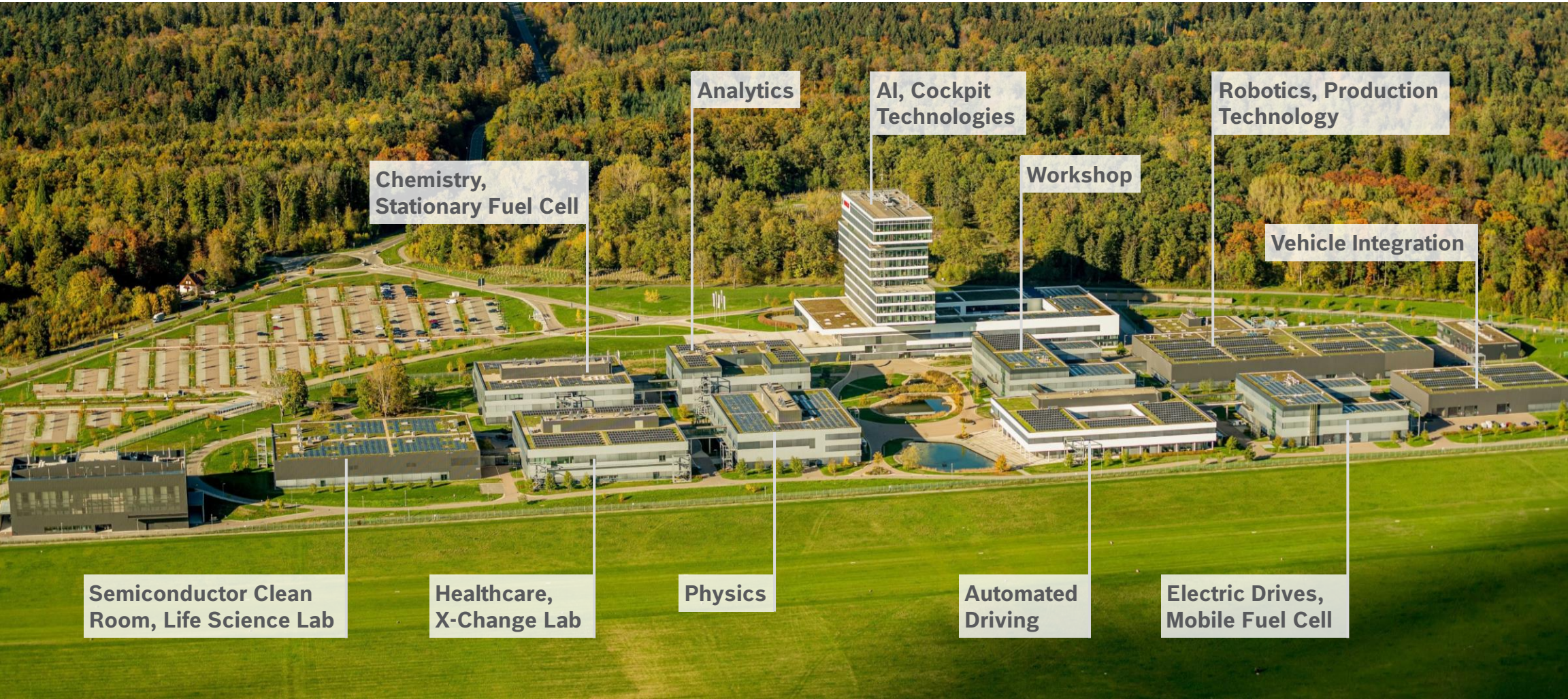





Research portfolio | Business portfolio

BOSCH

# Bosch Research: Campus Renningen
## Facilities



# Bosch Research: Campus Renningen

## The best research facilities for the best research results

Clean Room

Robotics Research Lab

Anechoic Chamber

Test Track

Smart Life Lab

Modern Workspaces





BOSCH

# Robotics Products by Bosch

## Indego

- Only robotic mower with intelligent navigation
- Since 2013


## Roxxter

- SLAM-enabled vacuum robot
- Persistent mapping, annotations via App
- Telepresence
- Since 2019


## ActiveShuttle

- Award-winning AGV/AMR
- Transportation of stacks of boxes
- SLAM-enabled, fleet control
- Since 2020


## ROKIT Locator

- Award-winning cm-accurate SLAM SW
- For intralogistics and manufacturing
- Since 2019
- Further ROKIT products on the way

# Robotics Products by Bosch


## Smart Robotic Functions

- SW solutions & services by BEG for professional cleaning robots
- Obstacle avoidance, narrow space cleaning
- Since 2023

## Kassow Robots

- Collaborative robot portfolio
- 7-axis, strong, fast, simple
- Since 2022


## Smart Flex Effector

- Sensor-supported compliance module in 6 dofs
- Very accurate and sensitive manipulation, e.g., peg-in-hole tasks
- Since 2022

## Smart Item Picking

- Model-free manipulation for order picking and placing
- AI-based perception and manipulation skills
- Since 2022

# 02

# Quality of SW-intensive Products: Theoretical Background

# Software Engineering for Reliable Autonomous Robots
## Motivation

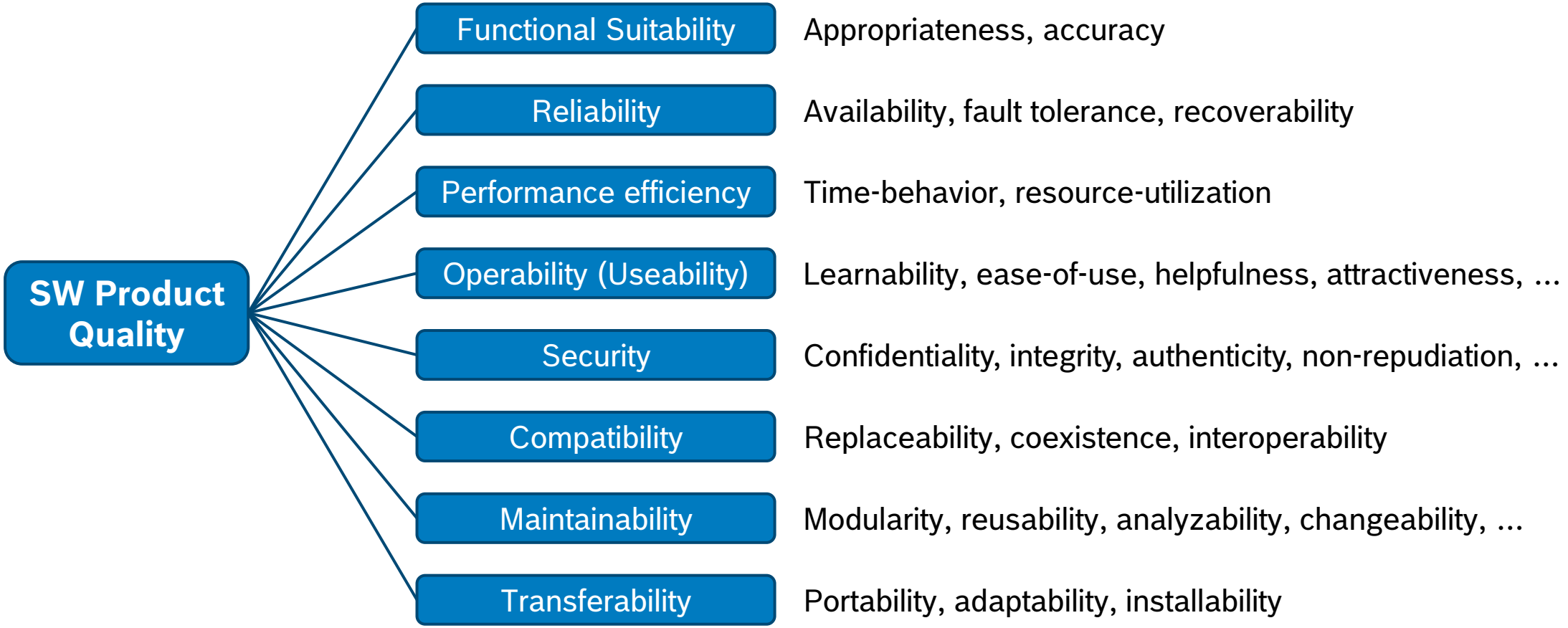
Key characteristics of robotics SW engineering:

- Very complex products – from sensor hardware to cloud
- Embedded systems (with varying depth)
- Need for robot autonomy in unknown environments
- Small development teams (compared with automotive)

**How to create good products of high quality efficiently under these circumstances?**

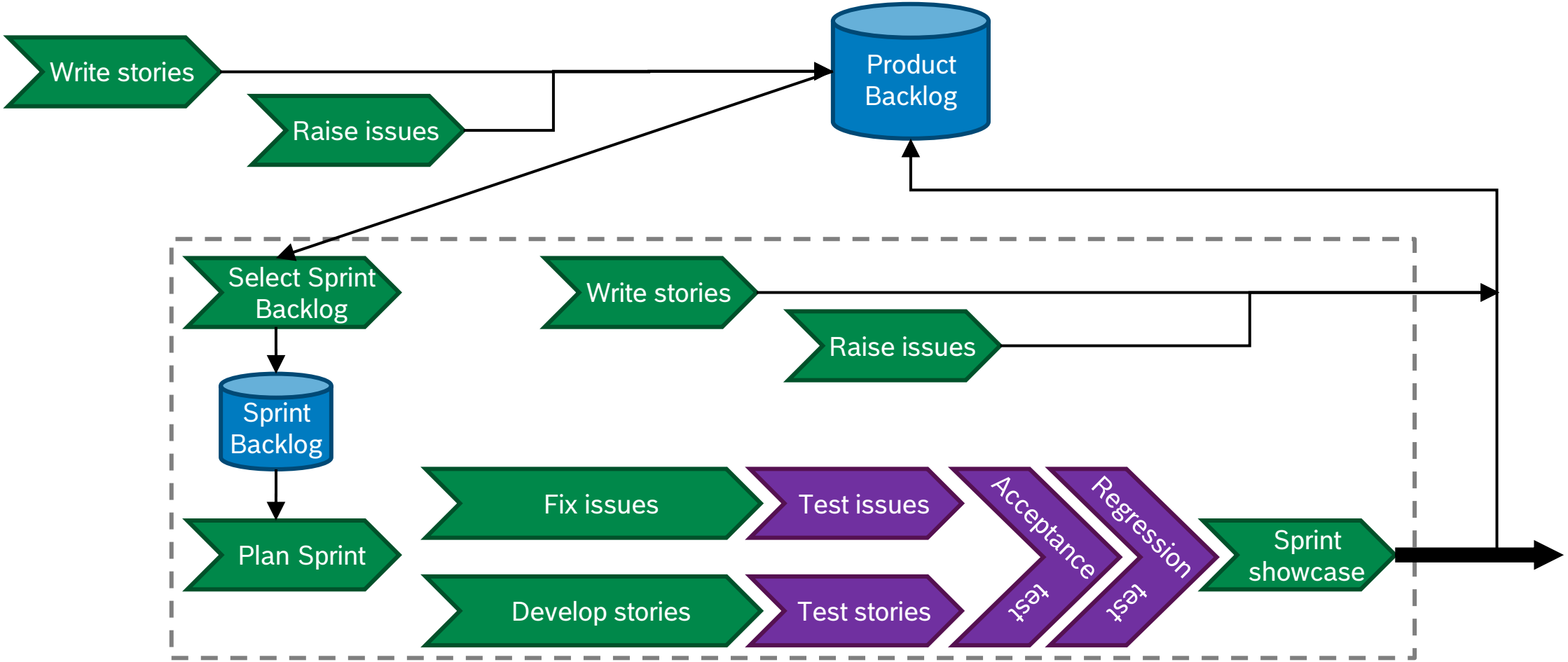
# Software Engineering for Reliable Autonomous Robots

## ISO 25010 – Quality has many Facets



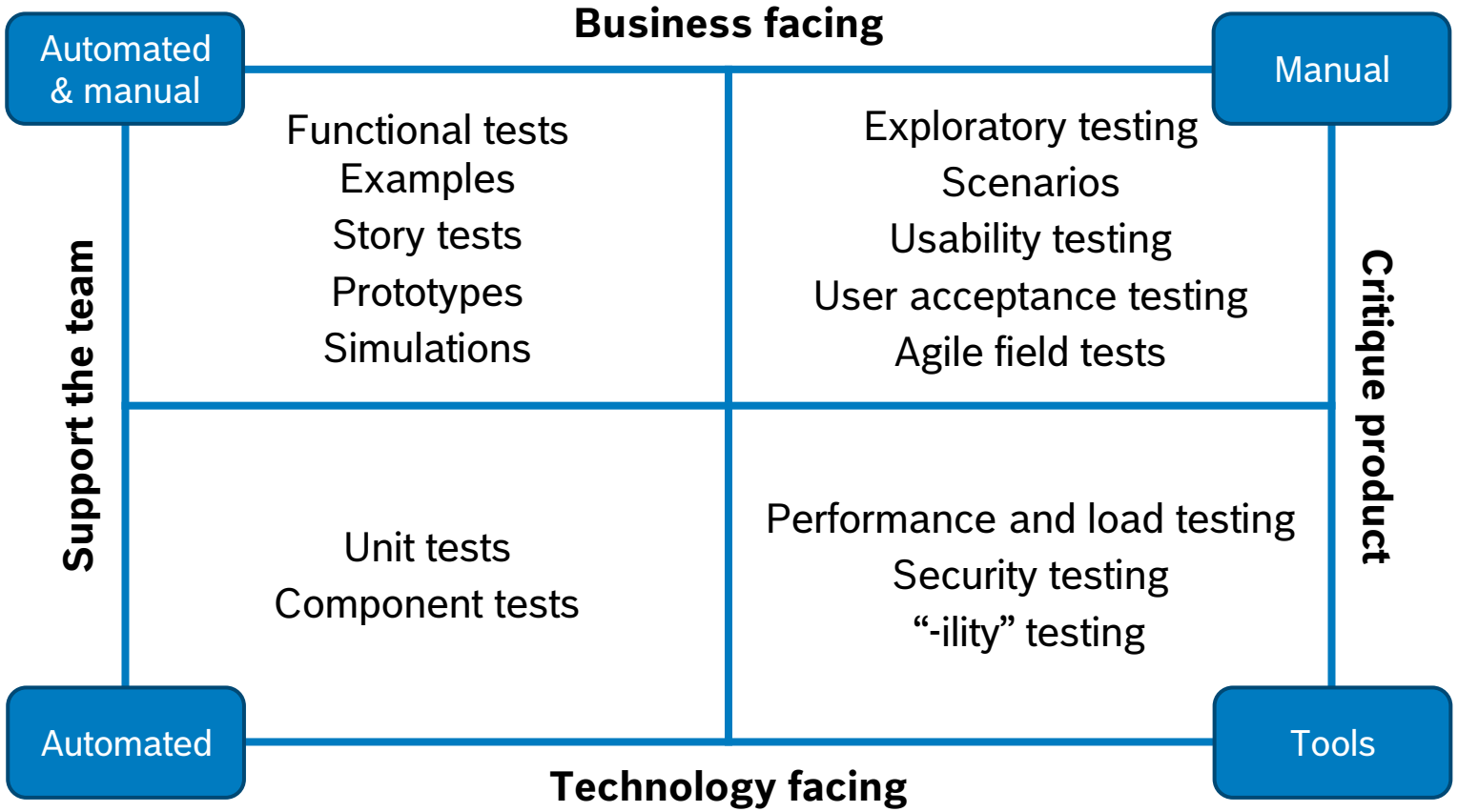
# Software Engineering for Reliable Autonomous Robots

## Quality Assurance in Sprint Cycle according to ISO 29119-1



# Software Engineering for Reliable Autonomous Robots

## Agile Test Quadrants (Crispin & Gregory, 2008)







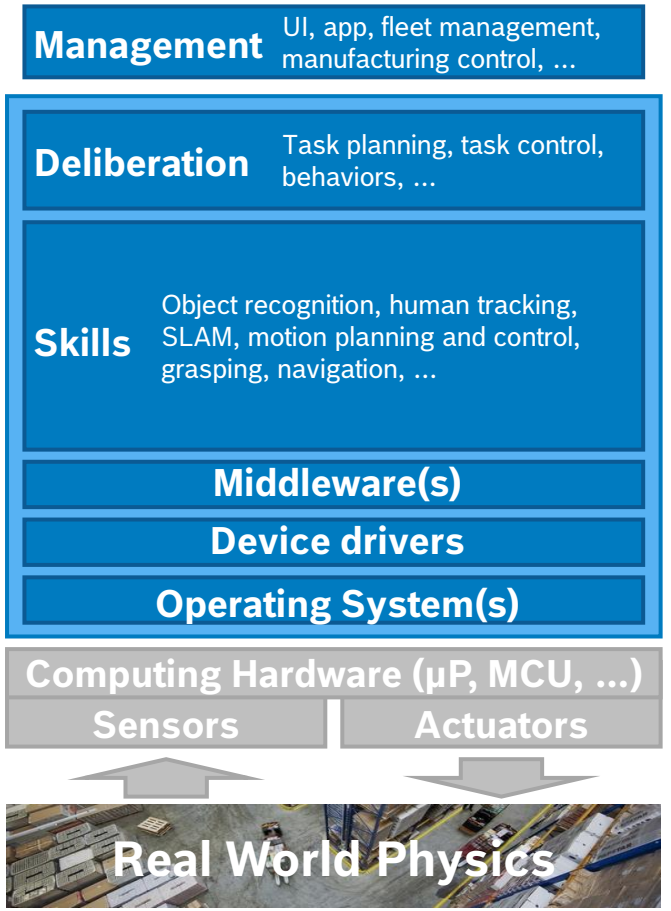
BOSCH

# 03

# Learnings and Best Practices along the typical Robot SW Stack

# Software Engineering for Reliable Autonomous Robots

## A Very Generic Robot SW Stack



# Software Engineering for Reliable Autonomous Robots
## Microcontrollers

Robots are networks of microprocessors and microcontrollers → distributed systems

Typical challenges and issues

- Time-consuming programming
- Breaks in toolchains
- Hardware issues (cables, connectors, …)
- Specific debuggers
- Time stamping and synchronization

**Lessons learned and best practices**

- Develop on stronger compute platform first
- Use HW monitoring and diagnostics
- Invest in good time synchronization …
- … and precise timestamping of sensor data

micro-ROS Overview Concepts Tutorials API Blog Search via Lunr.js

# micro-ROS

micro-ROS puts ROS 2 onto microcontrollers

## Mission

Bridging the gap between resource-constrained microcontrollers and larger processors in robotic applications that are based on the Robot Operating System.


## Why Microcontrollers?

Microcontrollers are used in almost every robotic product. Typical reasons are:

- Hardware access
- Hard, low-latency real-time

## Key Features

✓ Microcontroller-optimized client API supporting all major ROS concepts
✓ Seamless integration with ROS 2
✓ Extremely resource-constrained but flexible middleware
✓ Multi-RTOS support with generic build system
✓ Permissive license
✓ Vibrant community and ecosystem
✓ Long-term maintainability and interoperability
✓ Natively integrated into Vulcanexus, the all-in-one ROS 2 tools set
✓ Much more...

## Architecture

The architecture of the micro-ROS stack follows the ROS 2 architecture. Dark blue components are developed specifically for micro-ROS. Light blue components are taken from the standard ROS 2 stack.


## Getting Started

Our tutorials and demos give you a quick start with micro-ROS. The basic tutorials can even be completed without a microcontroller.


## Commercial support

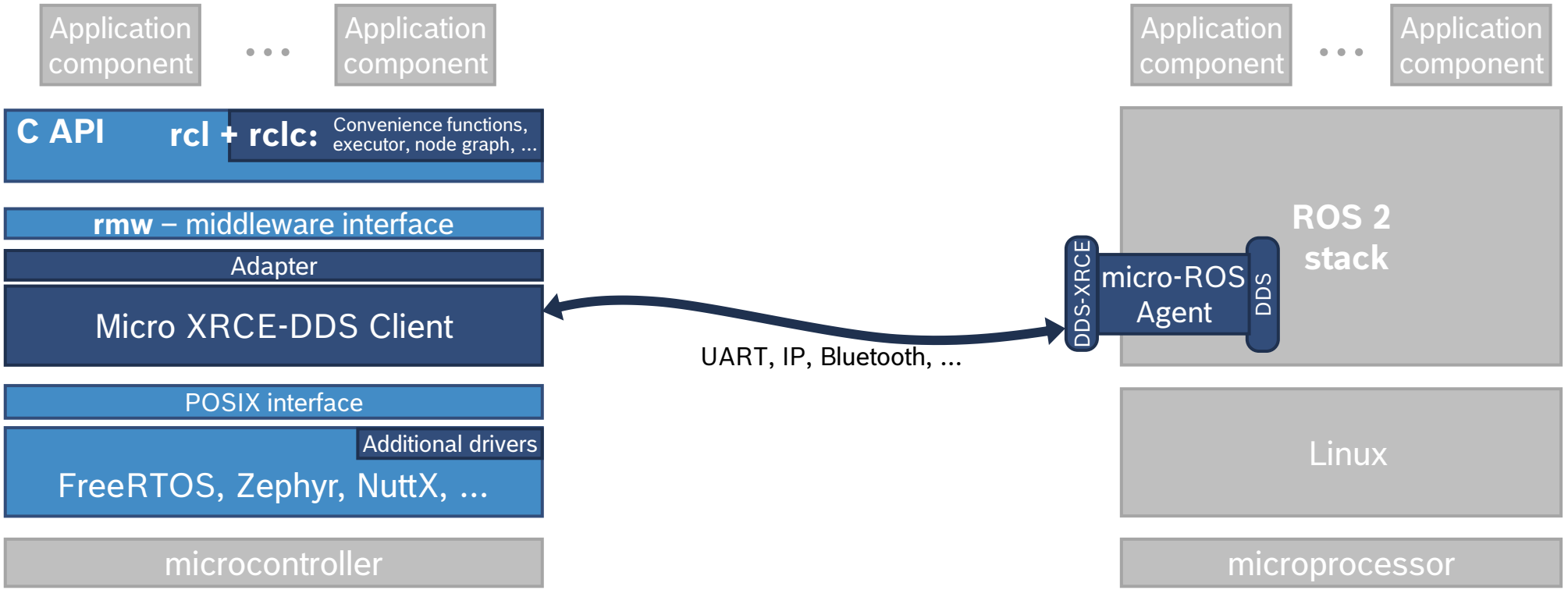
eProsima provides commercial support to boost micro-ROS projects:

✓ Port micro-ROS to your platform (HW, RTOS, transport)
✓ Efficient & reliable communication layer between µC and DDS Data Space (ROS 2)

**https://micro.ros.org/**

# Software Engineering for Reliable Autonomous Robots

## Micro-ROS




Kaiwalya Belsare, et al.: **Micro-ROS**. In: Robot Operating System (ROS): The Complete Reference (Volume 7), Springer, pp. 3–55, 2023.

# Software Engineering for Reliable Autonomous Robots
## Micro-ROS

Hardware support: **Renesas EK RA6M5 • RasPi Pico • Teensy • STM32 • ESP32 • …**

RTOS: **bare metal (Arduino) • Mbed OS • Zephyr • FreeRTOS • NuttX • …**

Middleware: **static memory pools • thread-safety • embedded RTPS RMW beta support**

Client library: **parameters • node lifecycle • ROS graph • services • diagnostics • executor**

Build systems: **STM32CubeMX/IDE • Arduino • ESP32 IDF • Zephyr • … • and ROS 2 Tooling**

# Software Engineering for Reliable Autonomous Robots
## Skills (Capabilities)

Most development-intensive layer.

Typical challenges and issues

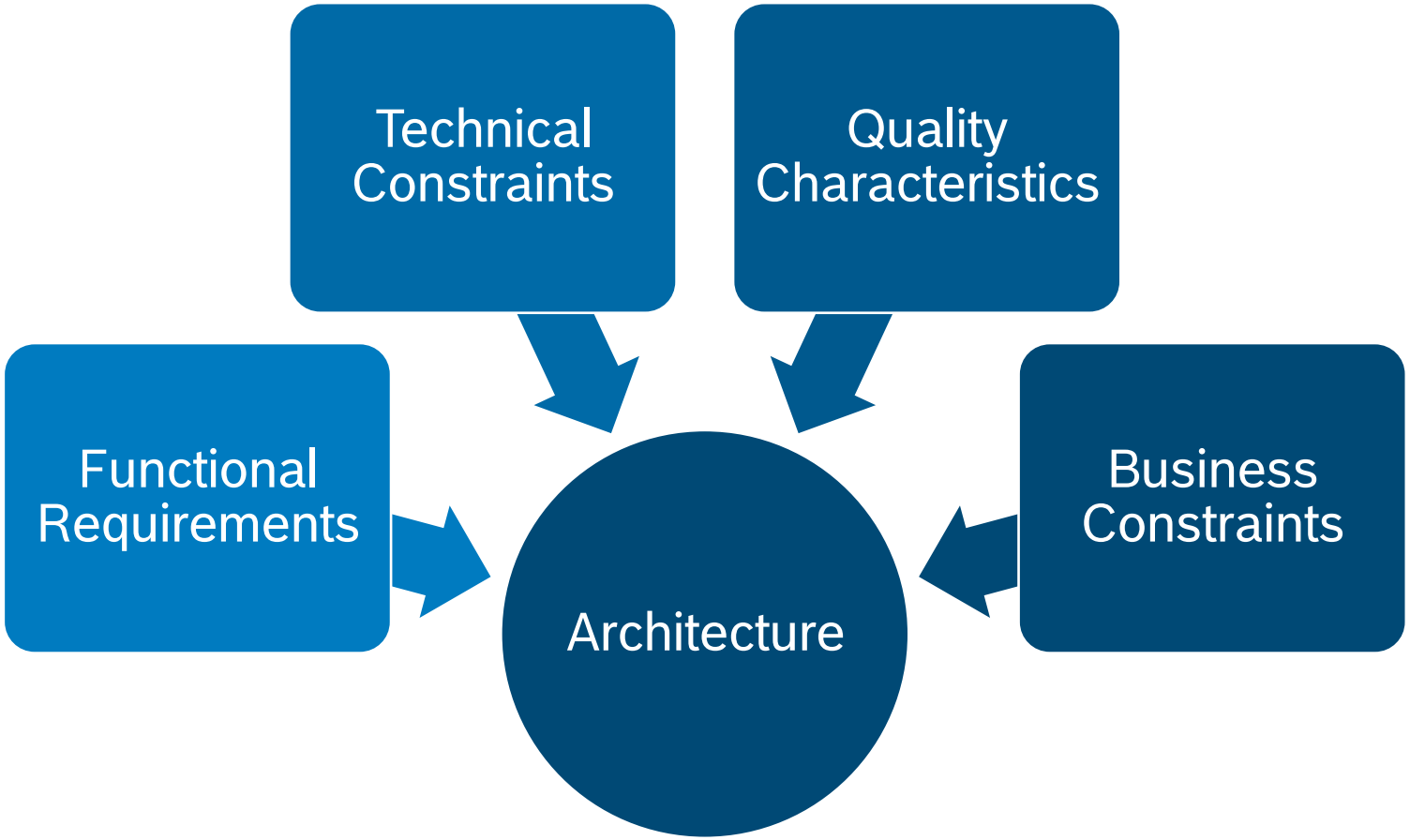
- Reuse of existing SW assets
- Modularization for such reuse and for parallelization of development
- High computational complexity
- Special demands on compute hardware
- Explainability and debuggability

**Lessons learned and best practices**

- Systematic scenario catalogs for requirements engineering
- Design as native libraries with separated ROS interfaces
  - for stand-alone testing and debugging
  - flexible integration of larger subsystems
- Regression and KPI testing
- Balanced strategy between component and system testing
- Explicit contingency and error handling

# Software Engineering for Reliable Autonomous Robots

## Architecture Drivers and Characteristics



# Software Engineering for Reliable Autonomous Robots
## Quality Scenarios and Architecture Fitness Functions

- Quality Scenarios „What do we need in our software to be successful?“

| Quality goals (Priority)<br>definition<br>What qualities shall be achieved with what priority? | Scenarios<br>What shall be achieved triggered by what in what circumstances? | Solution approach<br>How do you want to solve it? | Mandatory in Quality Level X | Technical risk<br>High, Medium, Low<br>Why? | Link | Basic test plan<br>How do you want to mitigate the risk (Analysis, tests, special reviews, ATAM, .. )? | Extended test plan<br>In case of a field release what do you want to do in addition? |
|---|---|---|---|---|---|---|---|

- Architecture fitness functions: *„Any mechanism that performs an objective integrity assessment of some architecture characteristic or combination of architecture characteristics.”* (“Evolutionary Architecture”, Ford et al, 2017)

# Software Engineering for Reliable Autonomous Robots
## Simulation

We all know the sim-to-real gap …

Typical challenges and issues

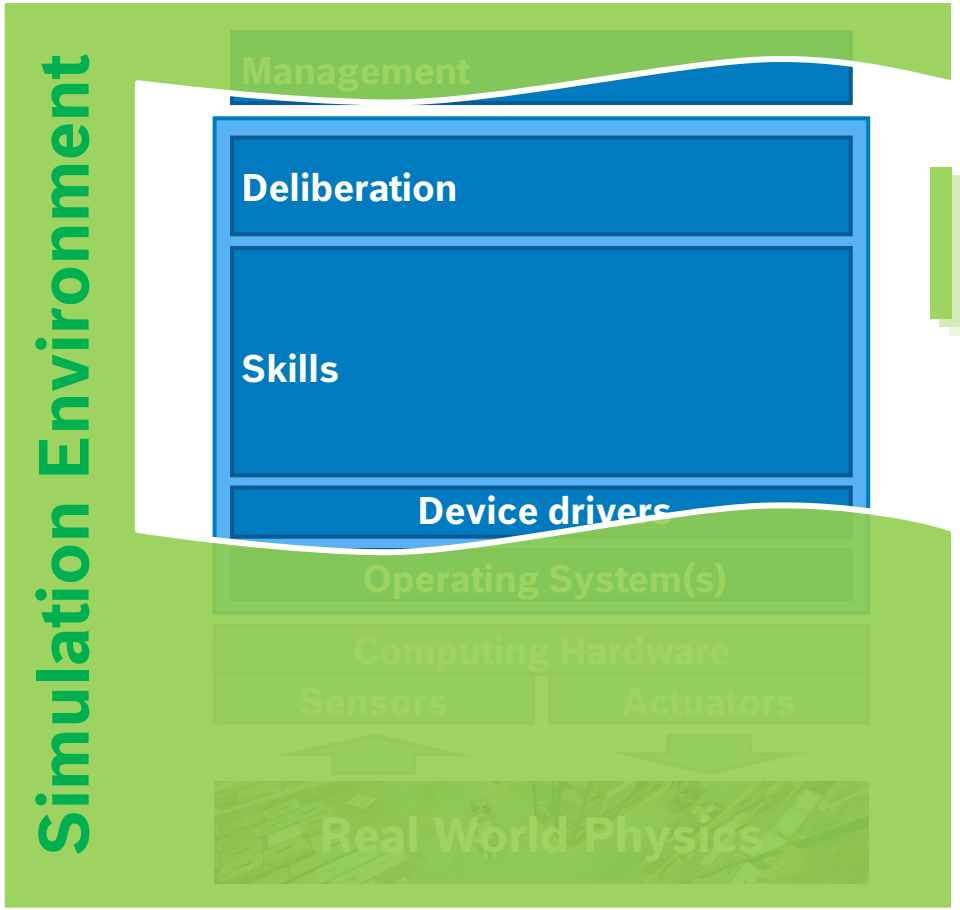
- Blind believe in simulation
- Efficient simulation
- Precision of simulation models
- Deterministic re-computation

**Lessons learned and best practices**

- Use specific simulation engines in addition to general 3D physics simulator
- Co-simulation (e.g., with FMI)
- Early real-world testing with prototypes
- Know the limits of your simulation models and simulators
- Combine simulation with real data
- Aim for deterministic simulation

# Software Engineering for Reliable Autonomous Robots

## Interfaces and Aspects of Typical 3D Physics System Simulation




Automation and monitoring

Scenario generation

Aggregation and analysis

Appropriate level of detail

... Use specific simulation environments

... Co-simulation

Computing platform effects

Sensor and actuator simulation

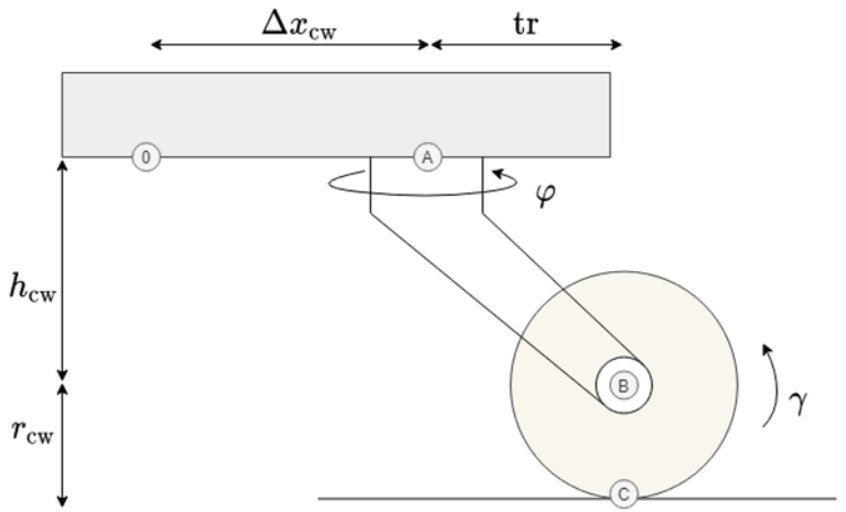
Robot and scene modeling

Rigid body dynamics, soft bodies, fluids, and agents




BOSCH

# Software Engineering for Reliable Autonomous Robots
## Limitations of Physics Engines: Bore Torque Effect

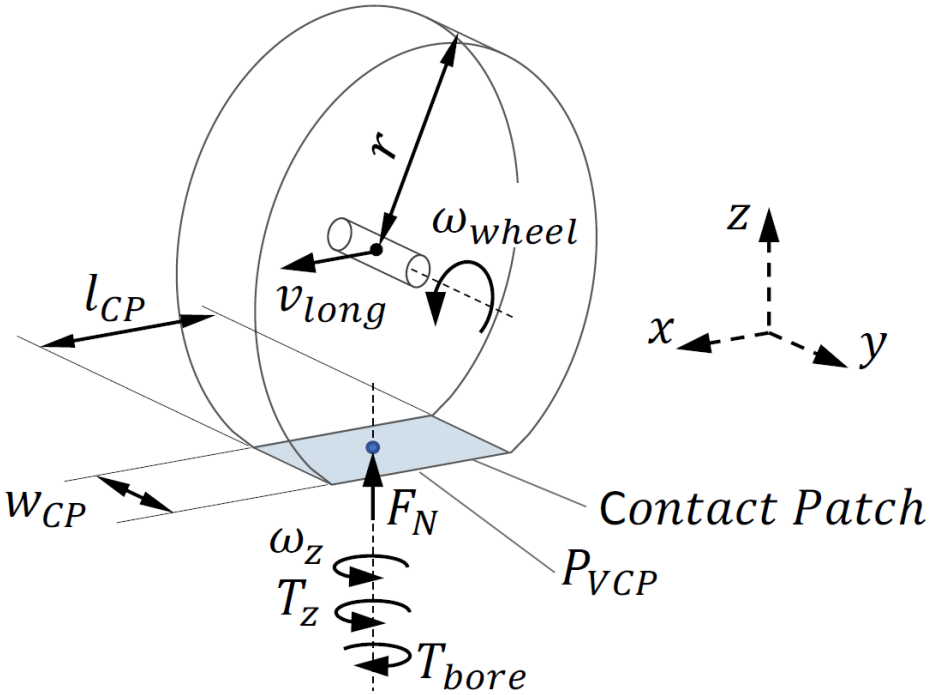



$$\dot{\varphi} = -\frac{1}{\mathrm{tr}} \cdot [(v - \omega \cdot \Delta y_{\mathrm{cw}}) \cdot \sin(\varphi) - (\omega \cdot \Delta x_{\mathrm{cw}}) \cdot \cos(\varphi)]$$

$$\dot{\gamma} = \frac{1}{r_{\mathrm{cw}}} \cdot [(v - \omega \cdot \Delta y_{\mathrm{cw}}) \cdot \cos(\varphi) + (\omega \cdot \Delta x_{\mathrm{cw}}) \cdot \sin(\varphi)]$$

# Software Engineering for Reliable Autonomous Robots

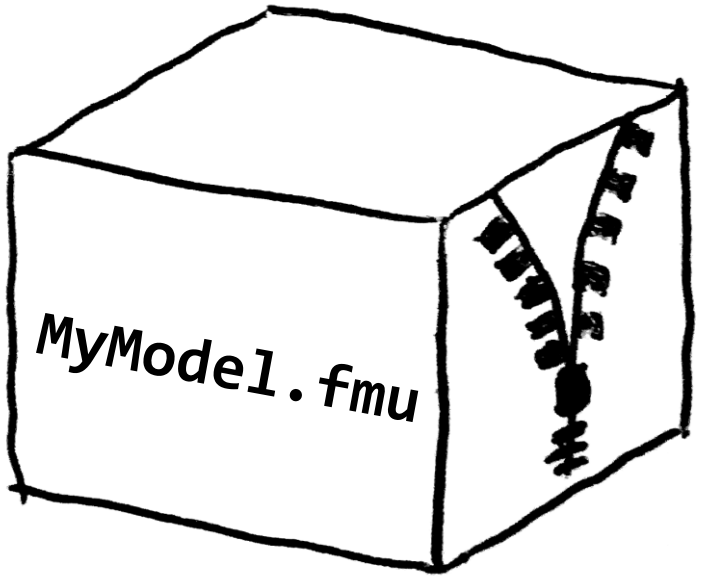
## Limitations of Physics Engines: Bore Torque Effect



# Software Engineering for Reliable Autonomous Robots
## Co-Simulation with Functional Mock-Up Units

Functional Mock-Up Interface

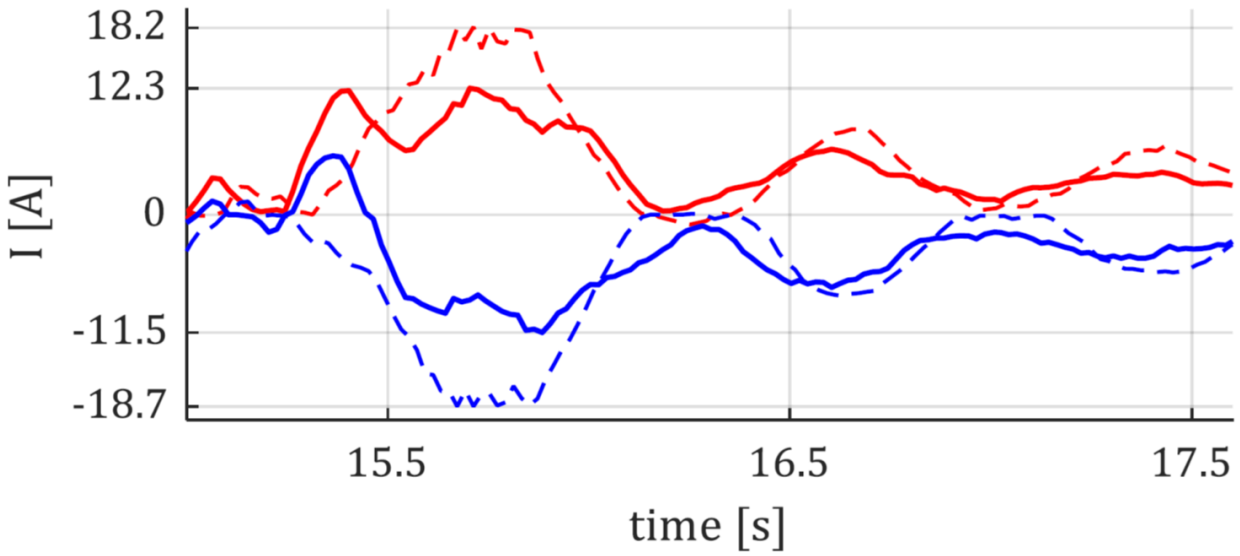
http://fmi-standard.org/




- Shared library
  - Equations
  - Solver
- modelDescription.xml
- *Optional: C sources*

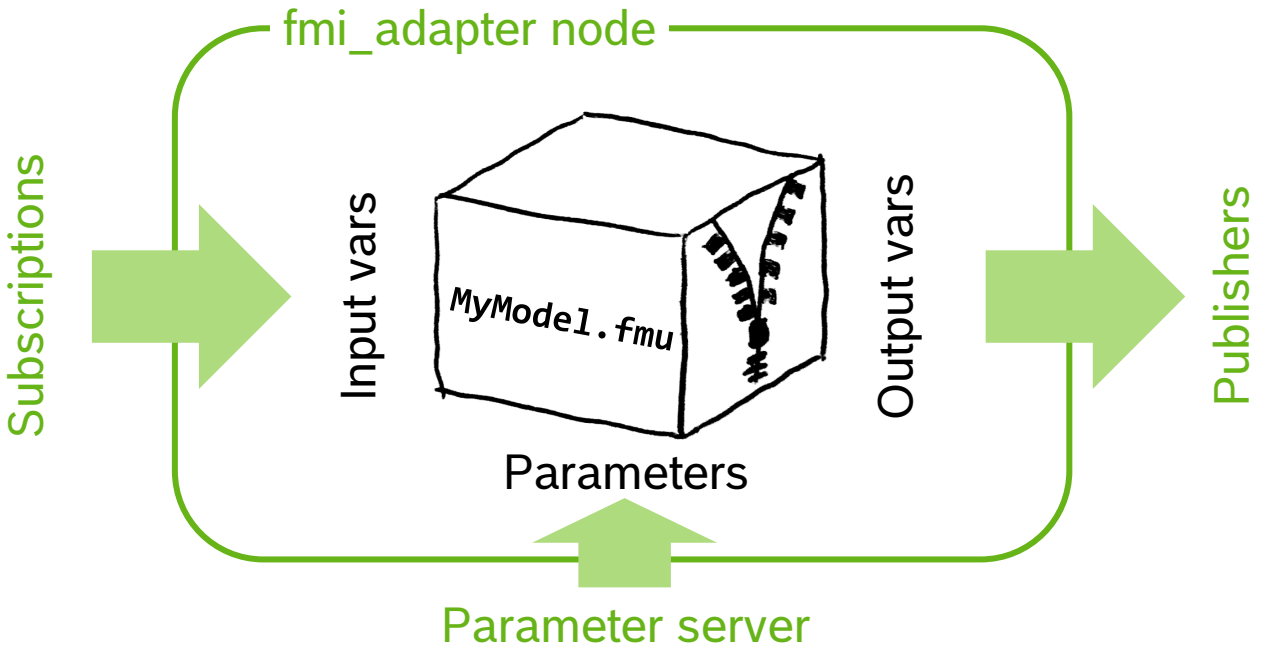
# Software Engineering for Reliable Autonomous Robots

## Simulating and Understanding Bore Torque Effects




Nikolas Schröder, Oliver Lenord, and Ralph Lange: **Enhanced Motion Control of a Self-Driving Vehicle Using Modelica, FMI and ROS.** Modelica Conference, 2019.

# Software Engineering for Reliable Autonomous Robots
## FMI-Adapter for ROS




```
$ ros2 launch fmi_adapter fmi_adapter_node.launch.py fmu_path:=[PathToFMUFile]
```

Files

rolling

Go to file t

- .github
- fmi_adapter
  - doc
    - high-level_architecture.pptx
    - high-level_architecture_with_a...
    - high-level_architecture_with_f...
  - include/fmi_adapter
  - launch
  - src
  - test
  - CHANGELOG.rst
  - CMakeLists.txt
  - README.md
  - package.xml
- fmi_adapter_examples
- 3rd-party-licenses.txt
- CONTRIBUTING.md
- CPPLINT.cfg
- LICENSE
- NOTICE
- README.md
- codecov.yml
- dependencies.repos



# The fmi_adapter package

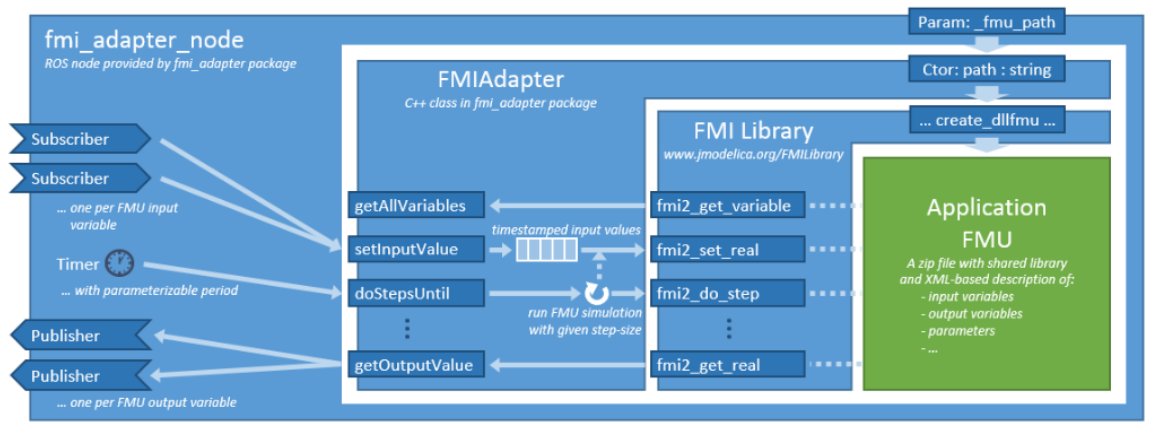
fmi_adapter is a small ROS 2 package for wrapping *functional mockup units (FMUs)* for co-simulation of physical models into ROS nodes. FMUs are defined in the FMI standard. Currently, this package supports co-simulation FMUs according to the FMI 2.0 standard only.

FMUs can be created with a variety of modeling and simulation tools. Examples are Dymola, MATLAB/Simulink, OpenModelica, SimulationX, and Wolfram System Modeler.

Technically, a co-simulation FMU is a zip file (with suffix .fmu) containing a physical model and the corresponding solver as a shared library together with an XML file describing the inputs, outputs and parameters of the model and details of the solver configuration. An addition, the zip file may contain the source code of the model and solver in the C programming language.

## fmi_adapter_node

fmi_adapter provides a ROS node *fmi_adapter_node* (class FMIAdapterNode derived from LifecycleNode), which takes an FMU and creates subscribers and publishers for the input and output variables of the FMU, respectively. Then, it runs the FMU's solver with a user-definable update period. This approach is illustrated in the following diagram.

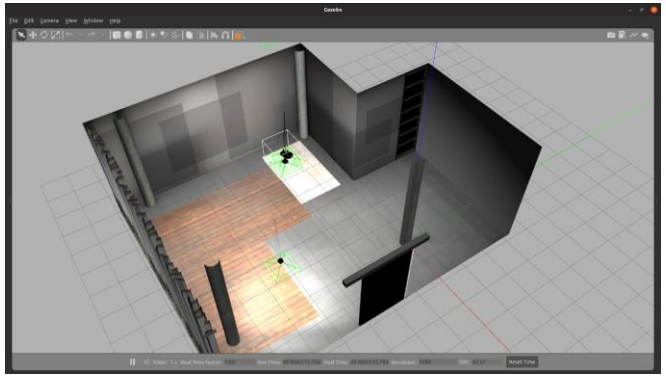



The fmi_adapter_node also searches for counterparts for each FMU parameter and variable in the ROS node parameters and initializes the FMU correspondingly.

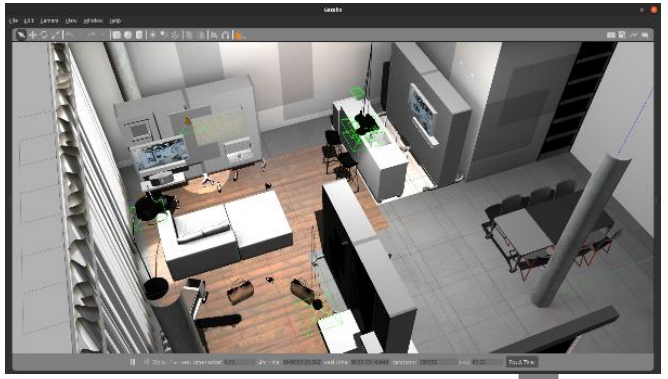
For this purpose, this package provide a launch file with argument *fmu_path*. Simply call

**https://github.com/boschresearch/fmi_adapter**

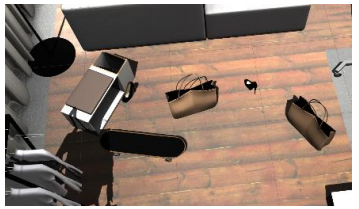
# Software Engineering for Reliable Autonomous Robots
## Generation and Randomization of Simulated Worlds

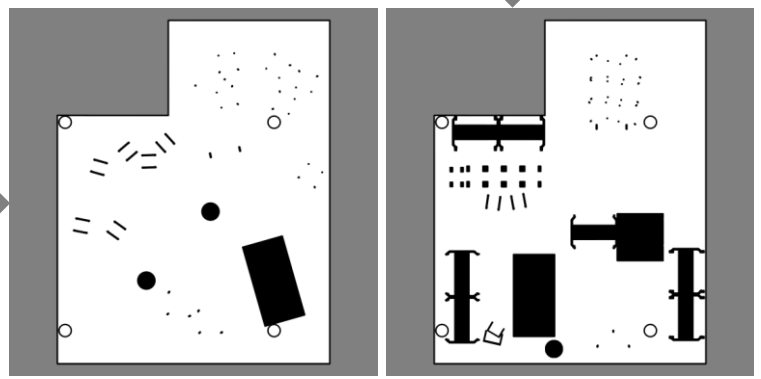


Empty room



Partial randomization for skateboard, shoe and bag models





Musa Marcusso: **pcg_gazebo_pkgs: A Python library for scripting and rapid-prototyping of simulated Gazebo models and worlds**. Talk at ROSCon 2019. vimeo.com/378683294.

# Software Engineering for Reliable Autonomous Robots
## Deliberation (Decision Making)

The most complex layer since everything comes together:

- Intended tasks
- Contingencies
- System-level errors

**Lessons learned and best practices**

- Separation between operational modes and deliberation
- Carefully choose between preprogramming, planning, and learning
- Explainable environment representation …
  - but accept additional skill-specific representations and inconsistencies
- Use of verification techniques

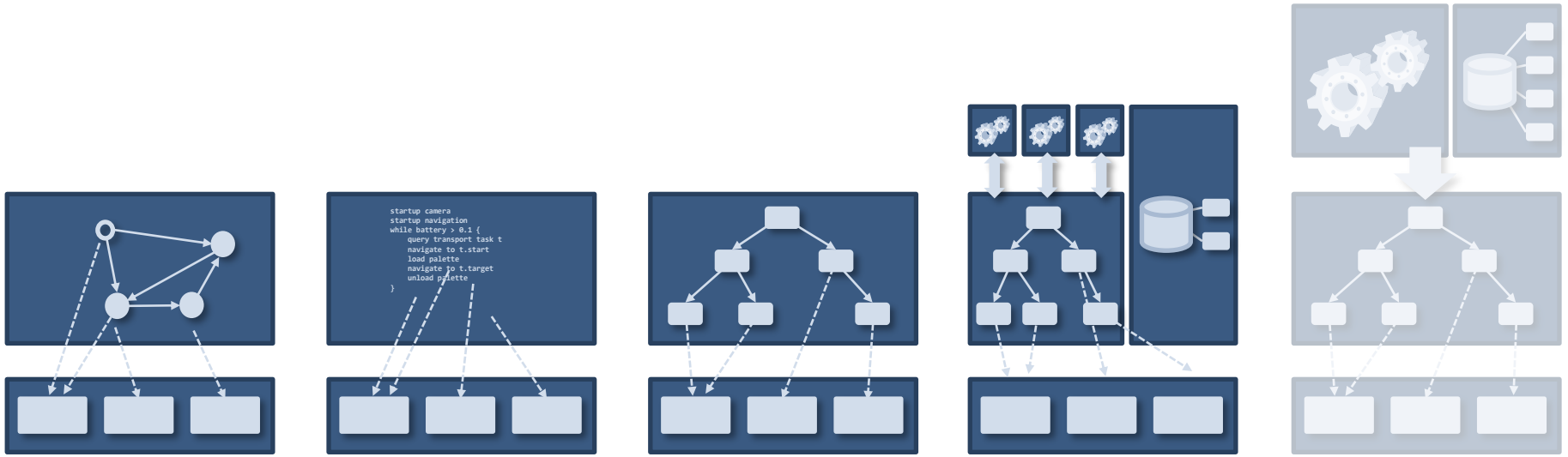
# Software Engineering for Reliable Autonomous Robots

## A nine-year-old slide …

Planning

Executive

Skills




Finite-state machine

Scripting language / DSL

Task/action hierarchy

Knowledge-based approach

Purely planning-based

- Name it “scene-graph-based” today
- Emphasize task-level learning!

ned

+ Fast
– Combinatorial explosion

Planning-based

+ Scalability
+ Simply recovering
+ Concise modeling
– Profound expertise required




BOSCH

Robotics DSL Zoo | Home | Subdomains | A&P Disciplines | Development Phases | Index | Contribute

# An Overview of Domain-Specific Languages in Robotics

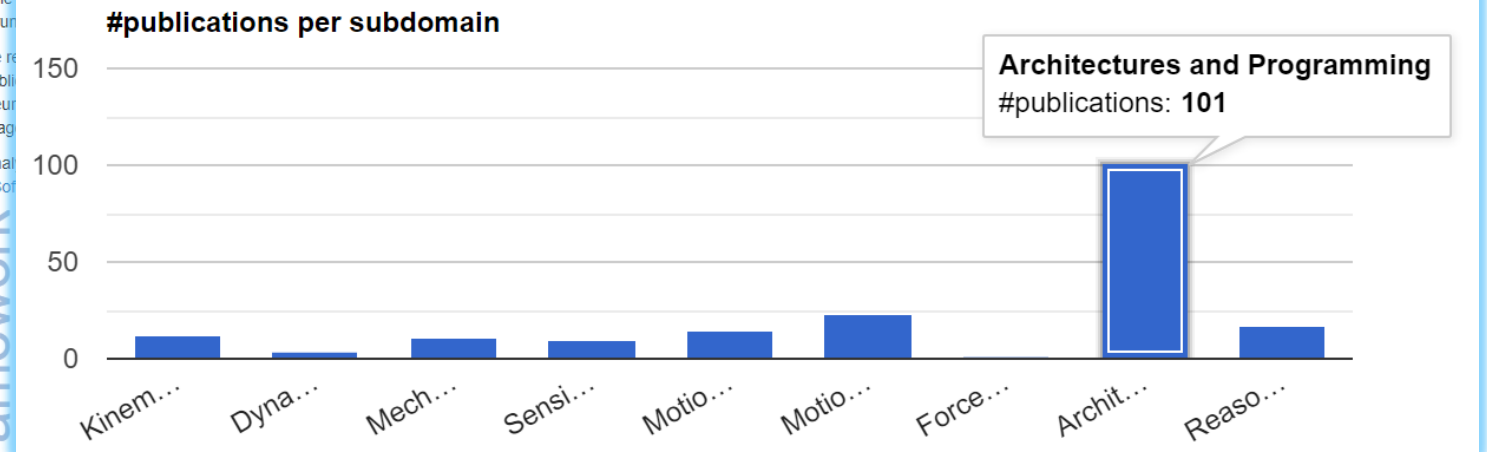
**initiated and maintained by A. Nordmann, N. Hochgeschwender , D. Wigand, and S. Wrede, updated on May 8th 2020**

This page provides an annotated bibliography of domain-specific languages in the area of robotics and automation technology facilitating the exchange between people interested in applying model-driven and domain-specific
languages (DSL)[1] idea and (on the long run

Practically, we aim to provide not only the re
links to documentation and software if publi
general, have been conducted by Van Deur
interchange of Eclipse/EMF-based languag

For a survey including a more detailed anal
and Languages in Robotics", Journal of Sof

componentbased description development framework flexible hybrid device english model




[1] A DSL in this context is a *"focussed, processable language for describing a specific concern when building a system in a specific domain. The abstractions and notations used are natural/suitable for the stakeholders who specify that particular concern"* [Voelter].

An important DSL is missing? Here is how to contribute! When using content of the Robotics DSL Zoo or referring to it, pleace consider citing our survey paper:

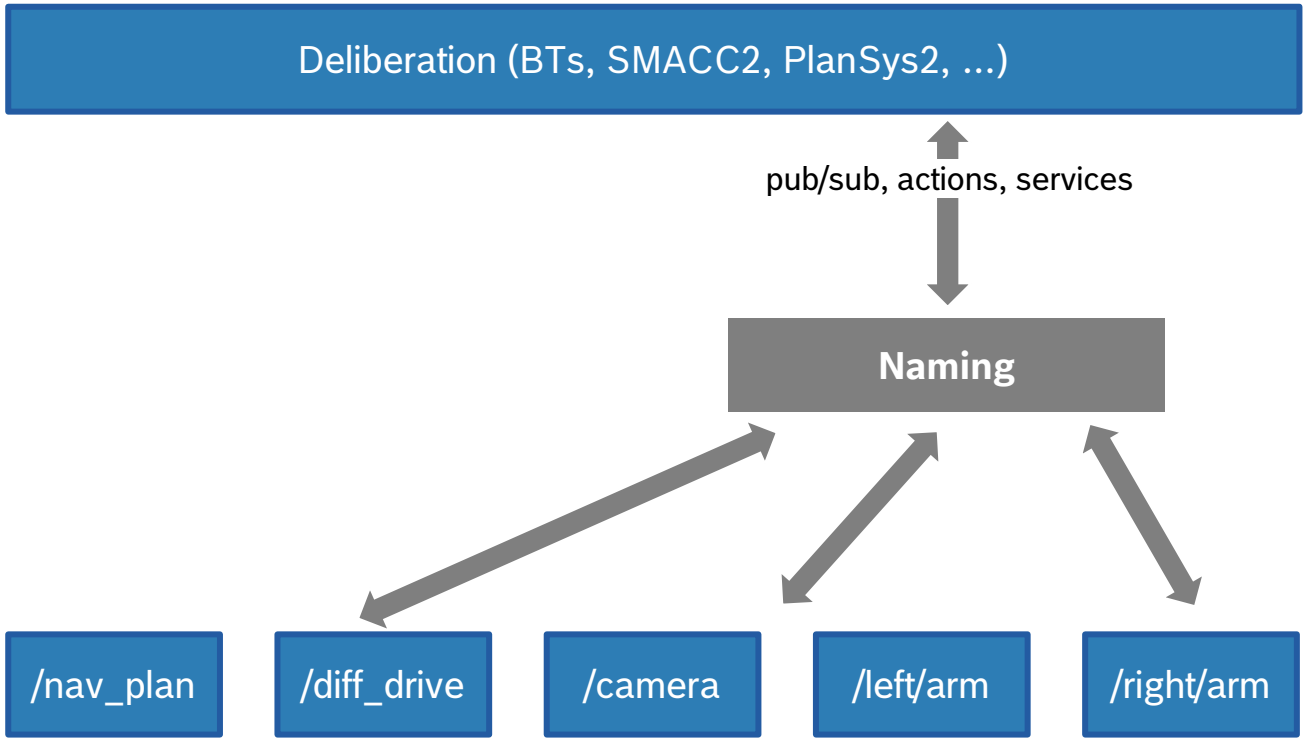
```
@article{nordmann2016survey,
  author       = {Nordmann, Arne and Hochgeschwender, Nico and Wigand, Dennis Leroy and Wrede, Sebastian},
  journal      = {Journal of Software Engineering in Robotics (JOSER)},
  number       = {1},
  pages        = {75--99},
  title        = {{A Survey on Domain-Specific Modeling and Languages in Robotics}},
  volume       = {7},
  year         = {2016},
}
```

**https://corlab.github.io/dslzoo/**

# Software Engineering for Reliable Autonomous Robots
## System Modes

Missing abstractions for:

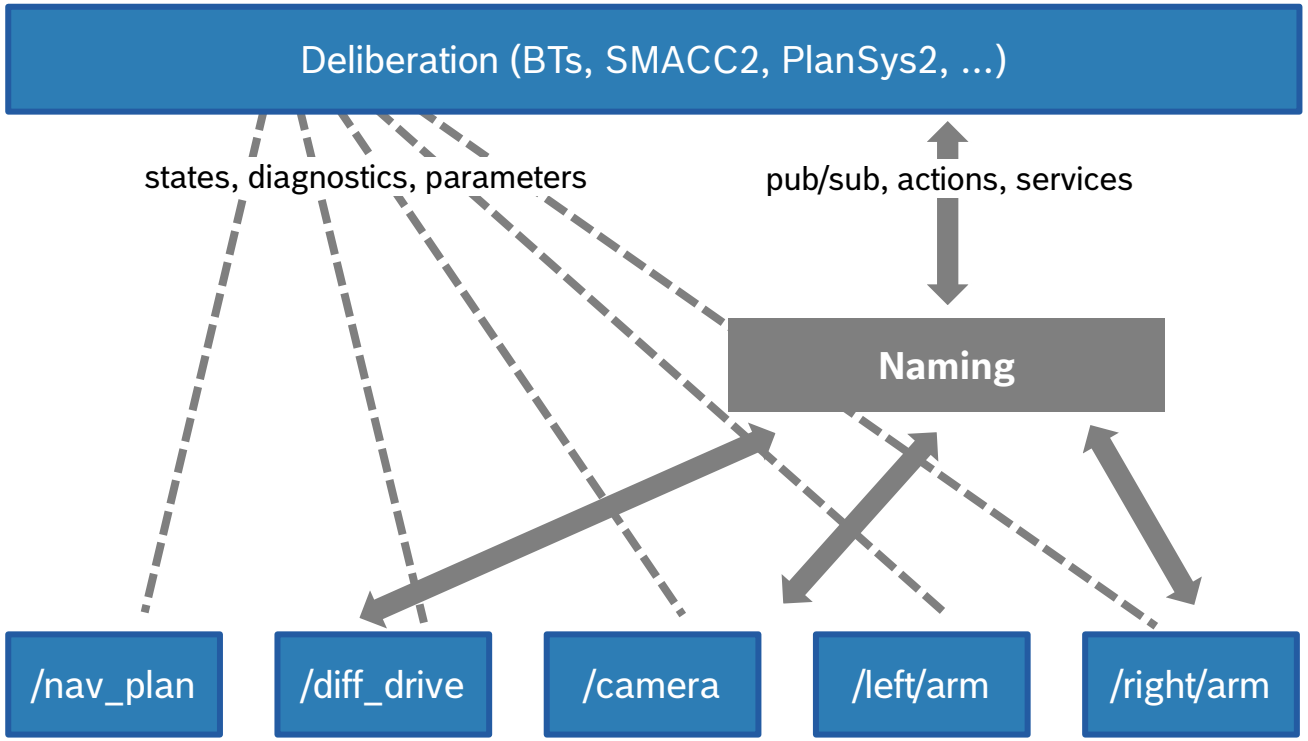
- States and lifecycle
- Parameters
- Diagnostics



# Software Engineering for Reliable Autonomous Robots
## System Modes

Missing abstractions for:

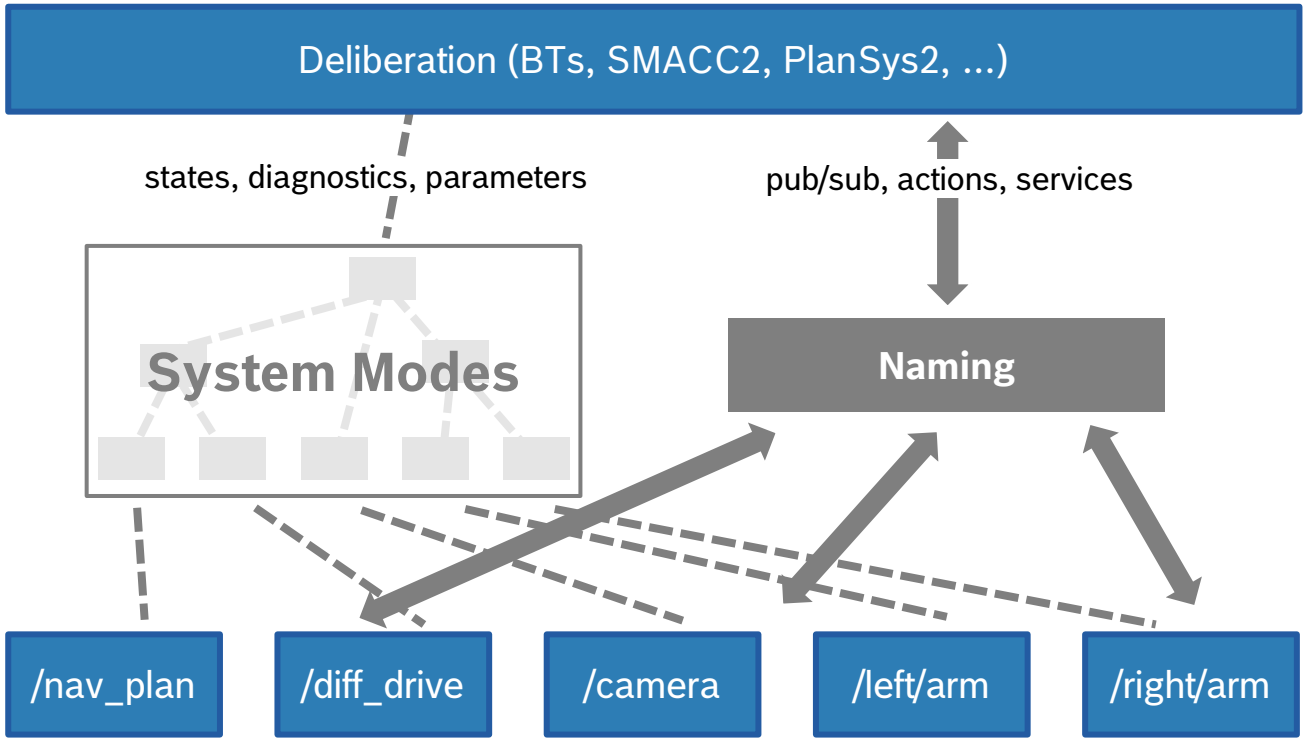
- States and lifecycle
- Parameters
- Diagnostics



# Software Engineering for Reliable Autonomous Robots
## System Modes

Missing abstractions for:

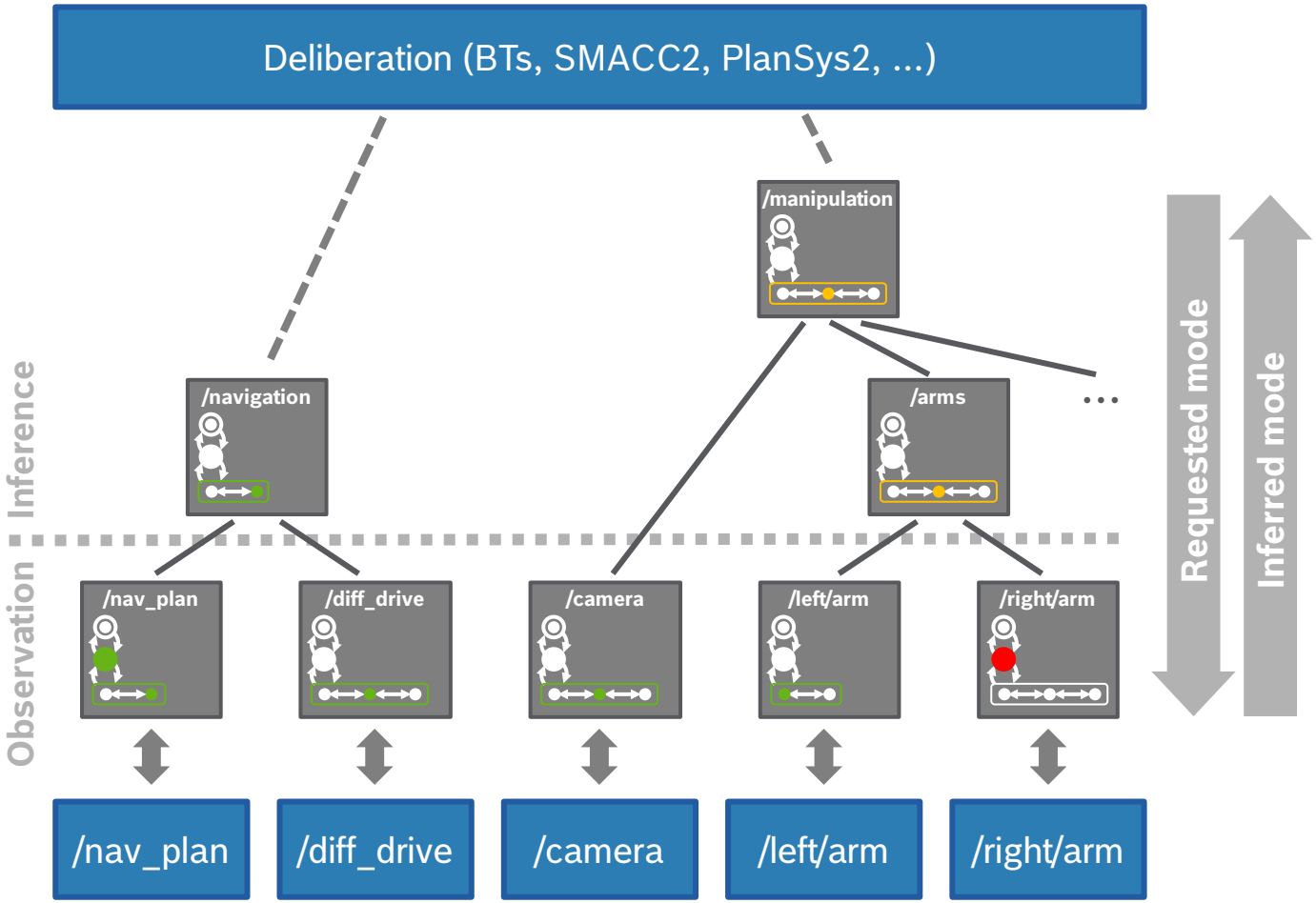
- States and lifecycle
- Parameters
- Diagnostics



# Software Engineering for Reliable Autonomous Robots

## System Modes

1. Modes for lifecycle nodes
   - Preconfigured parameter value sets
2. Hierarchy
   - Recursive grouping of nodes into (sub-)systems
   - Same lifecycle as nodes
3. Modes of (sub-)systems
   - Mapping to states + modes of their parts
   - Inference upwards along hierarchy




Ralph Lange and Arne Nordmann: **System Modes - model-based run-time state management of large systems.** ROSCon 2022. https://vimeo.com/767165876.

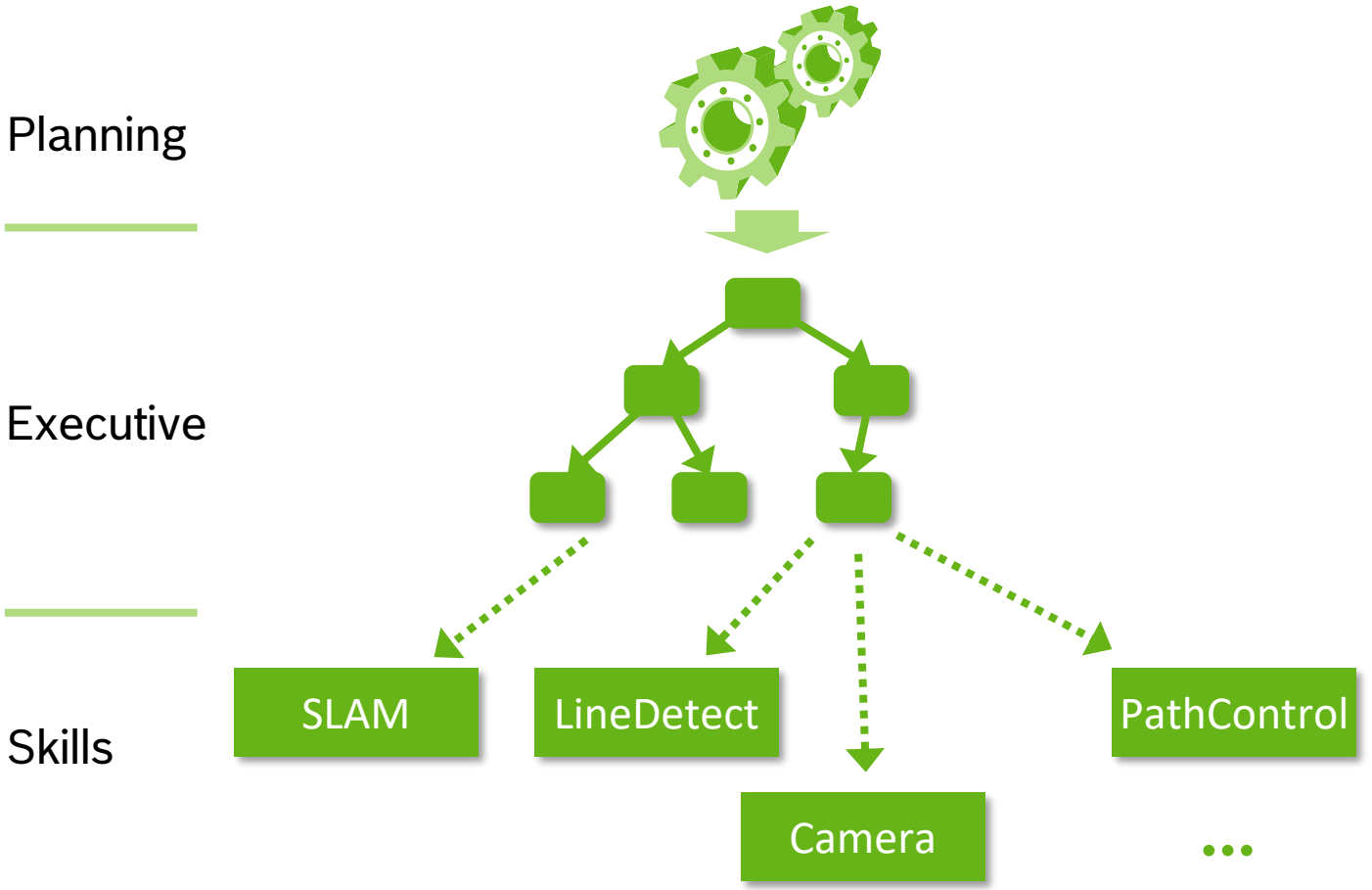
# 04

# Example for Classical Model-checking with Spin

Christian Heinzemann and Ralph Lange: **vTSL – A Formally Verifiable DSL for Specifying Robot Tasks**. IROS 2018.

# Software Engineering for Reliable Autonomous Robots

## vTSL: Motivation



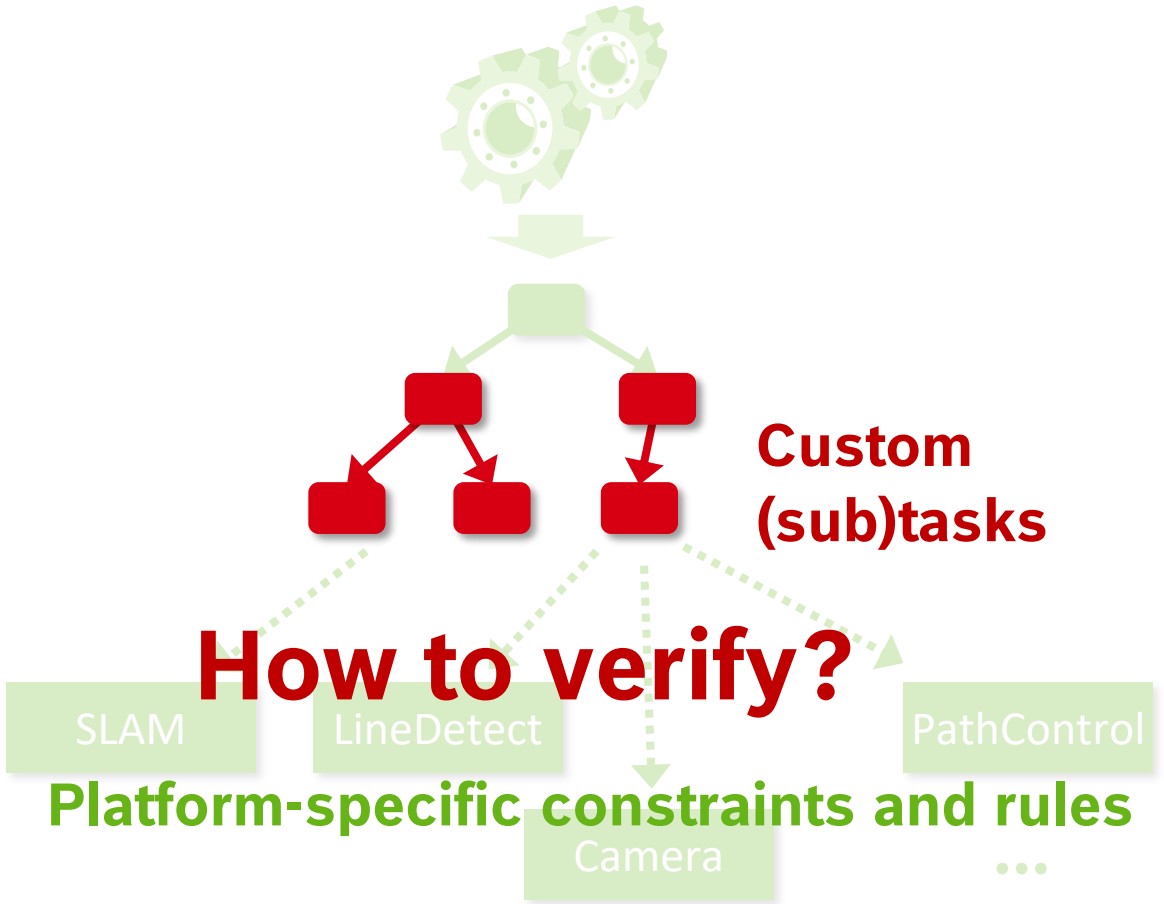





BOSCH

# Software Engineering for Reliable Autonomous Robots

## vTSL: Motivation

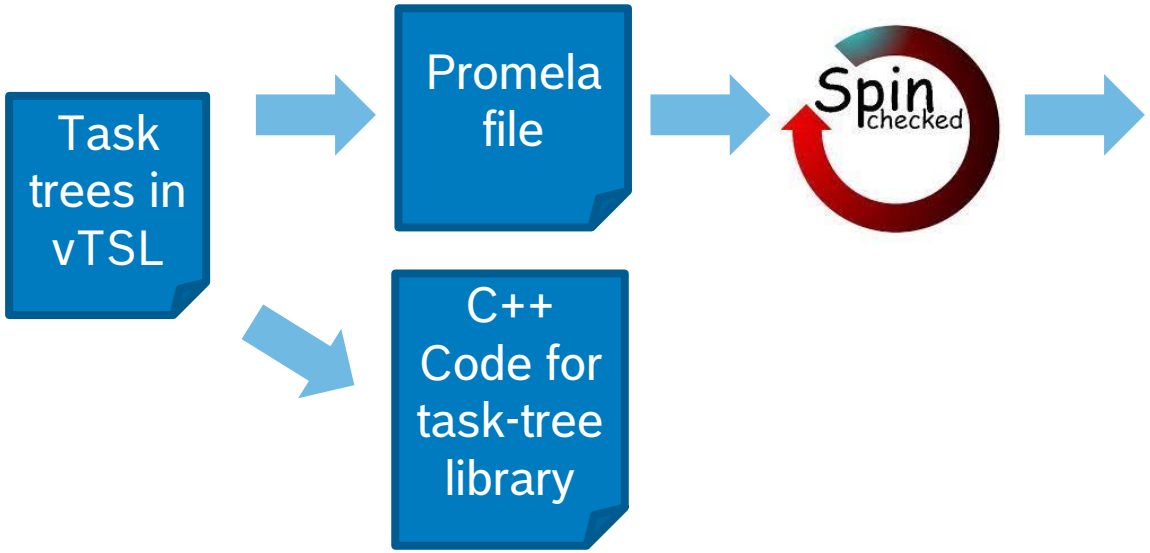







BOSCH

# Software Engineering for Reliable Autonomous Robots
## vTSL: Overview




```
$ spin -search -i MyTaskTree.pml
pan:1: assertion violated (BenchmarkComponent_simpleTopic.registeredClients<1) (at
depth 286)
pan: wrote MyTaskTree.pml.trail
pan: reducing search depth to 287

(Spin Version 6.4.8 -- 2 March 2018)
  + Partial Order Reduction

Full statespace search for:
  never claim          - (none specified)
  assertion violations  +
  cycle checks         - (disabled by -DSAFETY)
  invalid end states  +

State-vector 668 byte, depth reached 286, errors: 1
       97 states, stored
        7 states, matched
      104 transitions (= stored+matched)
      212 atomic steps
hash conflicts:         0 (resolved)

unreached in proctype vtsl__dispatcher
  MyTaskTree.pml:618, state 8, "abortRootAction = 0"
  MyTaskTree.pml:617, state 9, "abortSubaction[0] = rootActionPid"
  MyTaskTree.pml:624, state 16, "-end-"
  (3 of 16 states)
unreached in proctype vtsl__externalEventHandler
  MyTaskTree.pml:240, state 8, "vtsl__i = (vtsl__i+1)"
  MyTaskTree.pml:247, state 16, "vtsl__i = 0"

                ...

  MyTaskTree.pml:1112, state 106
  MyTaskTree.pml:1120, state 110, "-end-"
  (24 of 110 states)

pan: elapsed time 0.02 seconds
pan: rate      4850 states/second
```

# Software Engineering for Reliable Autonomous Robots

## vTSL: Language Concepts

### Verifiability

Fully automated transformation for verification tool – omit feature if not translatable

### Task tree semantics

Principle of decomposition as in Task Description Language or Hierarchical Task Networks

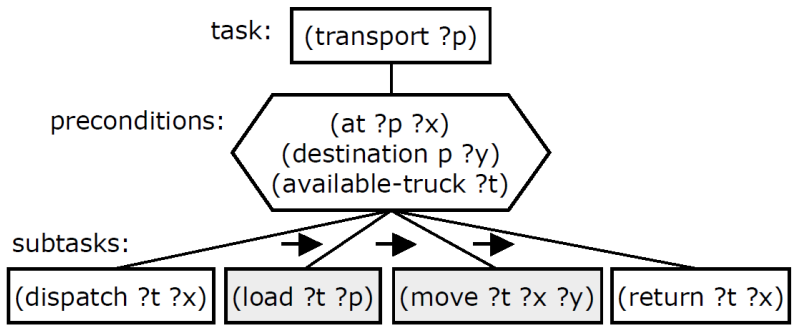



Image taken from Dana Nau et al.: "SHOP2: An HTN Planning System", *Journal of Artificial Intelligence Research*, Vol. 20, pp. 379-404, 2003.

### Expressiveness

Modern programming language features desired – resemblance with C/C++

### Synchronous Programming

Concurrency inspired by Céu, but not the strong determinism of synchronous languages

```
par/or do
    await RETRANSMIT;
with
    par/and do
        await 1min;
    with
        <send-beacon-packet>
    end
end
```

Snippet from Francisco Sant'Anna: "Structured Synchronous Reactive Programming with Céu", *Proc. of 14th MODULARITY*, pp. 29-40, 2015.

### Interfacing with skill layer

Interface with ROS and provide abstract replacement model for skill component behaviors

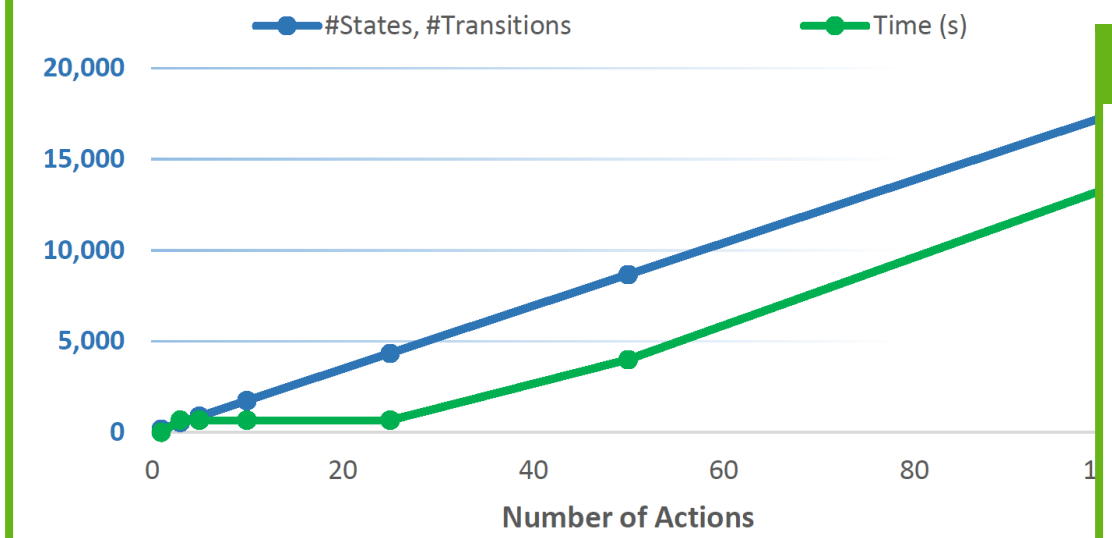
### Source code generation

Task trees should be directly generatable into implement-tation for the robot

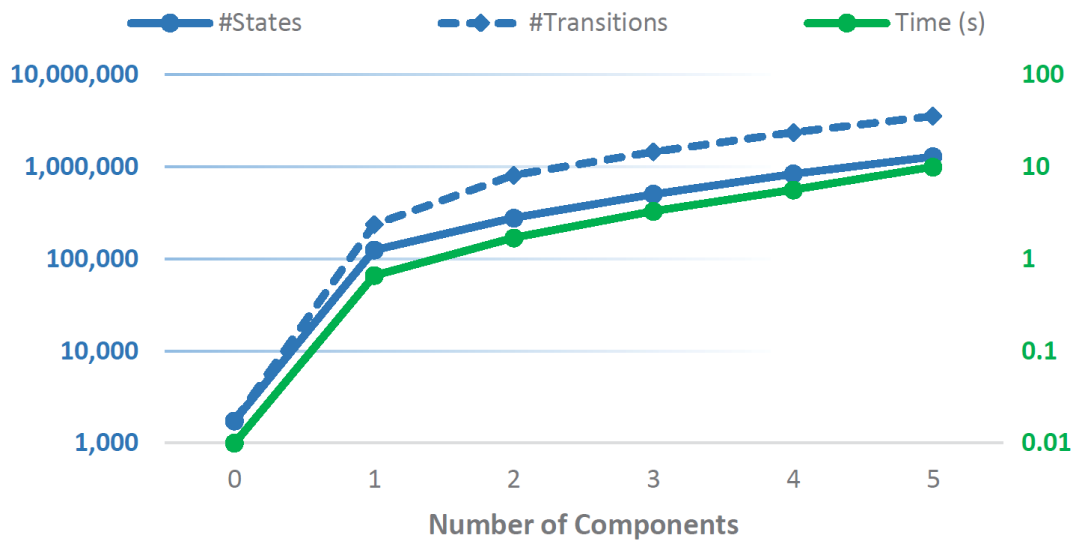
# Software Engineering for Reliable Autonomous Robots

## vTSL: Experiments

### Experiment 1: Scalability w.r.t. Tree Size




### Experiment 2: Scalability w.r.t. Component Count




**Limited scalability – in particular regarding number of components**

# 05

# The Why and How of ROS at Bosch

Ralph Lange: **From Early Research to Product Development – The Why and How of ROS at Bosch.** ROS-Industrial Conference 2022.



BOSCH

# 15 Years Journey with ROS

Bosch participates in PR2 beta program

Founding member of ROS-I Europe


Bosch sponsors development of ROS 2

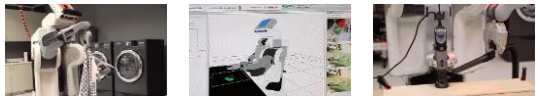
Founding member of ROS TSC

EU project micro-ROS

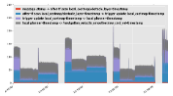
EU ITP MROS

EU project CONVINCE

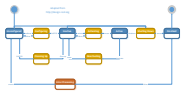
**2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024**



Tools and basic algorithms from PR2 beta program

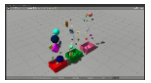


Tracepoints



Node lifecycle

rosbag2



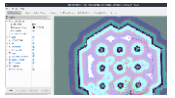
pcg_gazebo

Client library for C

ros2_control

System Modes

UUV simulator



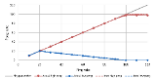
rviz2

Functional Mock-Up Interface

fmi_adapter

iceoryx

Middleware adapter



Refined Executor

Diagnostics

## From a small research team to hundreds of developers using ROS




BOSCH

# Classes of Use of ROS

Research and advance development

Development tool and environment

Middleware and framework in product software

# ROS connects Ecosystems

Bosch Products

ctrlX AUTOMATION

Tools, platforms

Simulink®

Unity®

Other domains

ROKIT Locator

ROS

Robotics


MCAP

fmi Functional Mock-Up Interface

Other domains




BOSCH

# Quality and Speed

- ROS 2 has paved way for professional use
  - Use of middleware standard DDS
  - Introduction of Quality Levels (REP-2004)
  - Much clearer OSS license documentation
- Not only high test coverage but also mileage

Working on automatic generation of copyright files for binary packages:
- github.com/boschresearch/ros_license_linter
- github.com/ros-infrastructure/rep/pull/347

- Fast proofs of concepts and MVPs
- Access to talents as well as development services

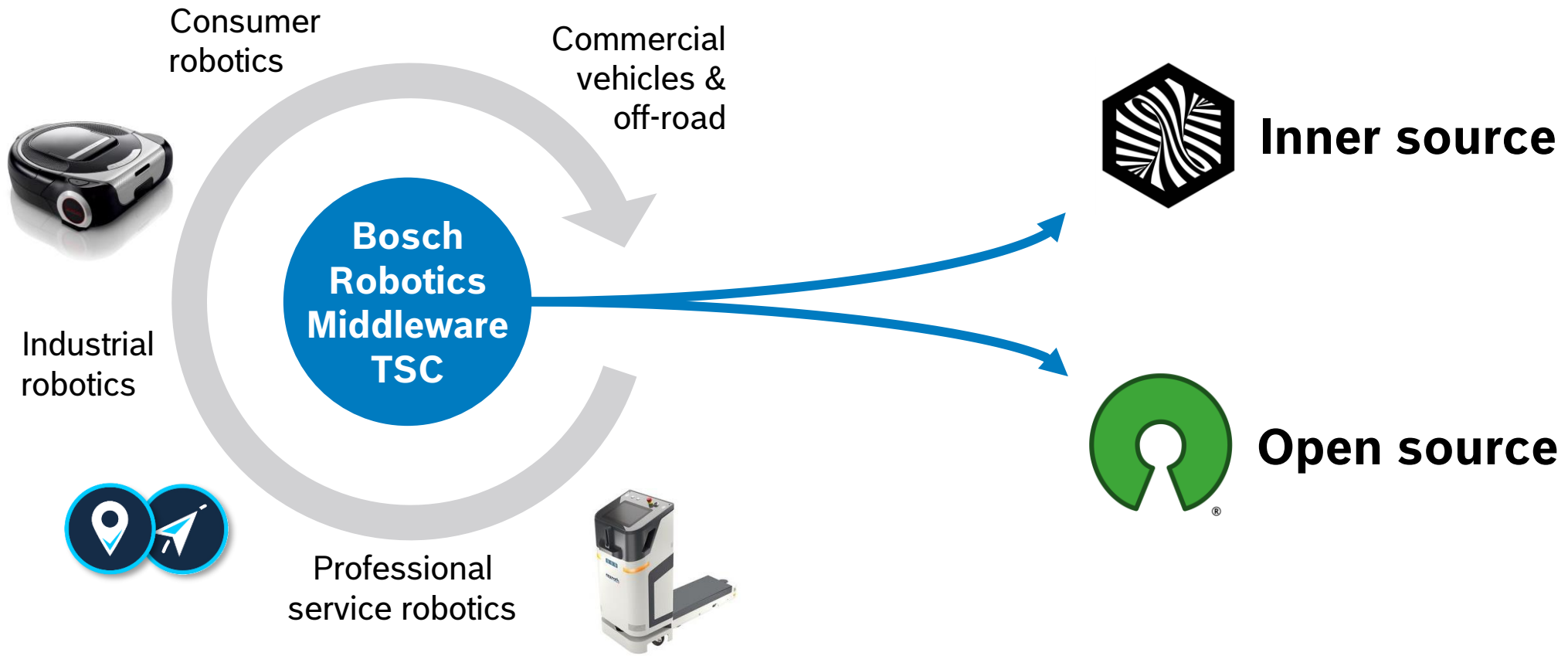
- Of course, there are also shady sides:
  - Issues in execution management and performance
  - Scattered and incomplete documentation

# Robotics Products by Bosch

- Heterogeneous use of ROS – all three classes
- Little to no use of ROS with legacy products
- Very different project phases
- Four business units involved



# Steering of Our ROS Strategy and Infrastructure





Ralph Lange (CR/AD CE-RSE) | 2022-12-15

Many thanks for your interest!

# Questions?

Michaela Klauck | michaela.klauck@de.bosch.com