Automatic Latency Management for ROS 2:
Benefits, Challenges, and Open Problems

T. Blass, A. Hamann, R. Lange, D. Ziegenbein, B. Brandenburg

SIC Saarland Informatics Campus

MAX PLANCK INSTITUTE FOR SOFTWARE SYSTEMS

BOSCH Invented for life

Our Goal: Apply real-time theory to robots

Rapid Development

ROS

Large open-source robotics framework & ecosystem

+

Timing Correctness

Real-Time Systems Theory

Abstracts from technical details → **Mismatch** ← Needs **detailed system model** to derive bounds

Tobias Blass

This Paper in a Nutshell

Why real-time theory is **difficult to apply** to ROS

Solution: The live latency manager **ROS-Llama**

- Collect information through **runtime introspection**
- **Automatically** configures real-time scheduler
- **Simple**, declarative specification

Evaluation: Case study on a TurtleBot 3

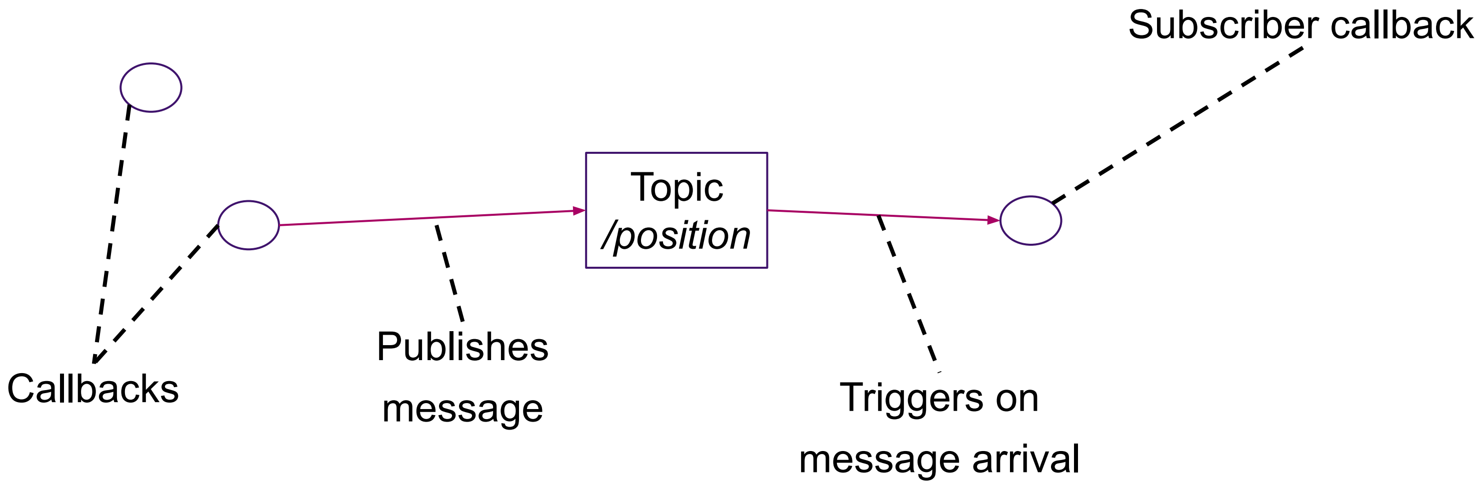
Tobias Blass

BOSCH

Outline

1. Background
2. Why is real-time theory difficult to apply to ROS?
3. The ROS Live Latency Manager (ROS-Llama)
4. Evaluation

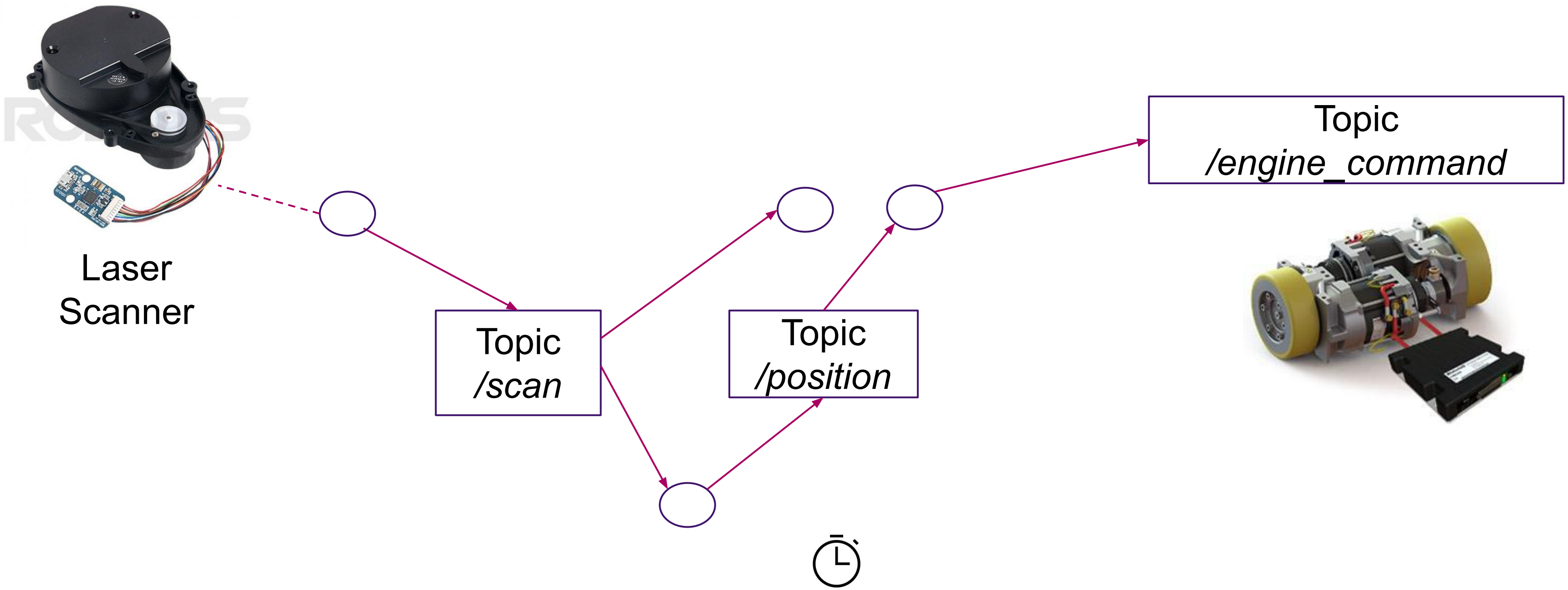
Background: ROS Pub-/Sub Mechanism



Tobias Blass

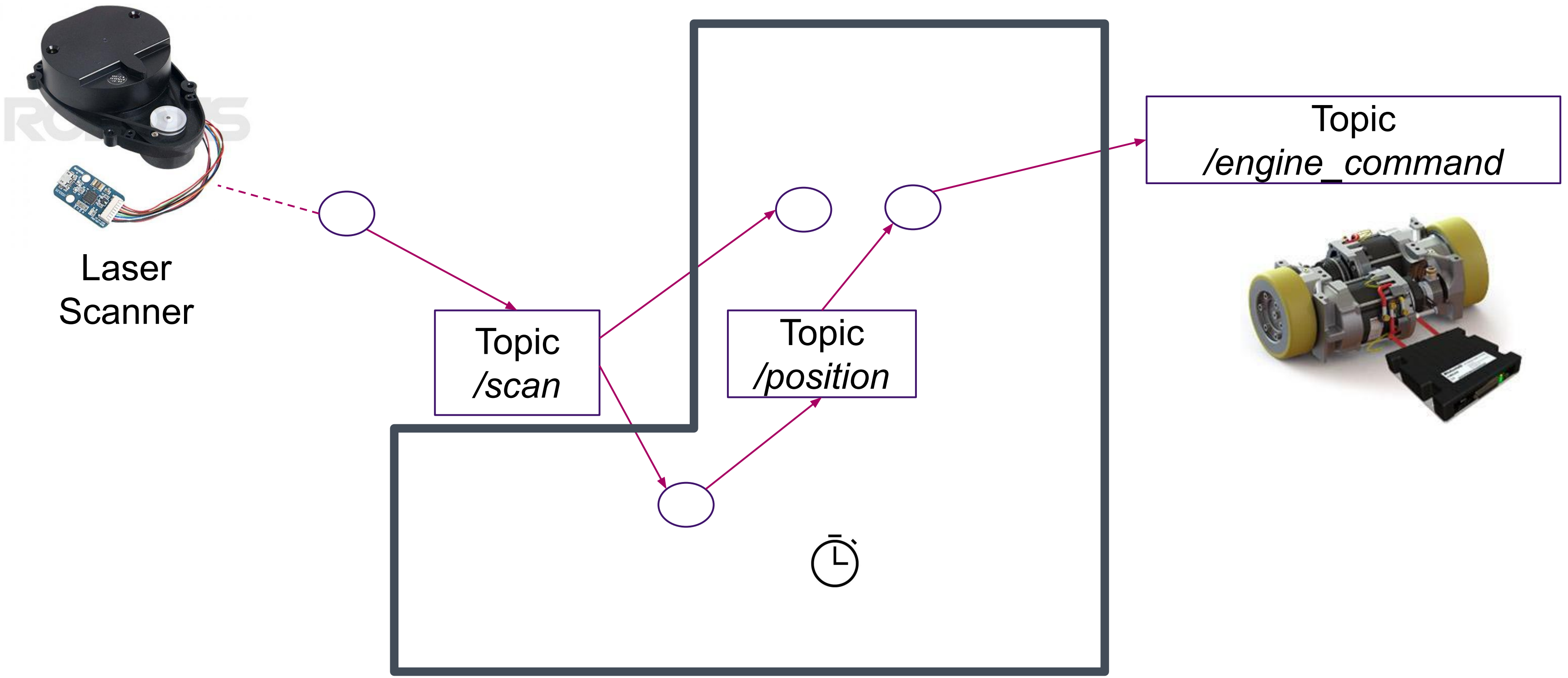
BOSCH

Background: Callbacks Form a Graph



Tobias Blass

Background: ROS Packages



Tobias Blass

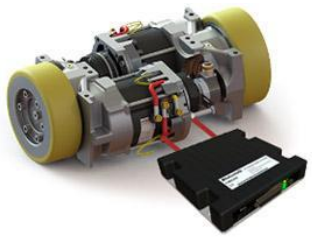
Background: ROS Packages

Laser Scanner

Topic
/scan

Navigation Package

Topic
/engine_command

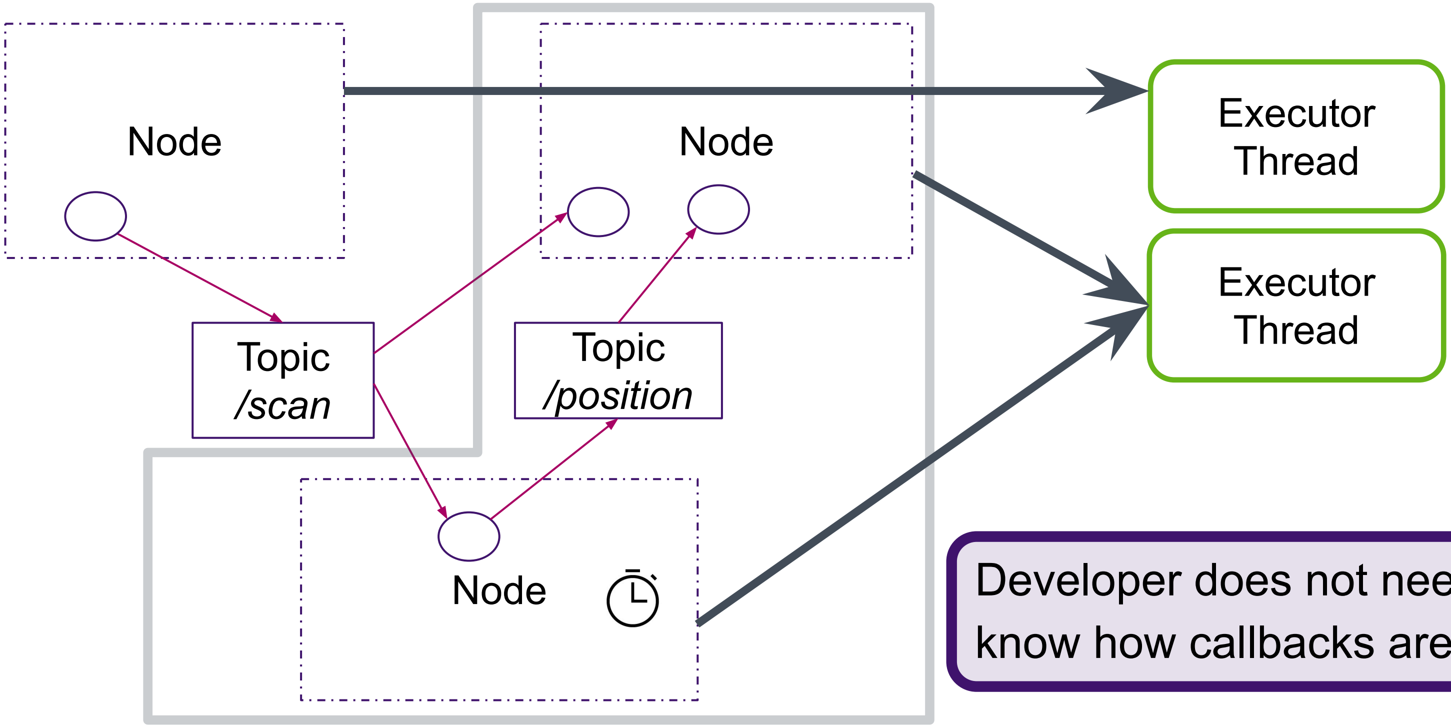


Developer does not need to know how the package works

Tobias Blass

BOSCH

Background: Callback Execution



Developer does not need to know how callbacks are run

Tobias Blass

BOSCH

Outline

1. Background
2. Why is real-time theory difficult to apply to ROS?
3. The ROS Live Latency Manager (ROS-Llama)
4. Evaluation

BOSCH

Why is real-time theory difficult to apply to ROS?

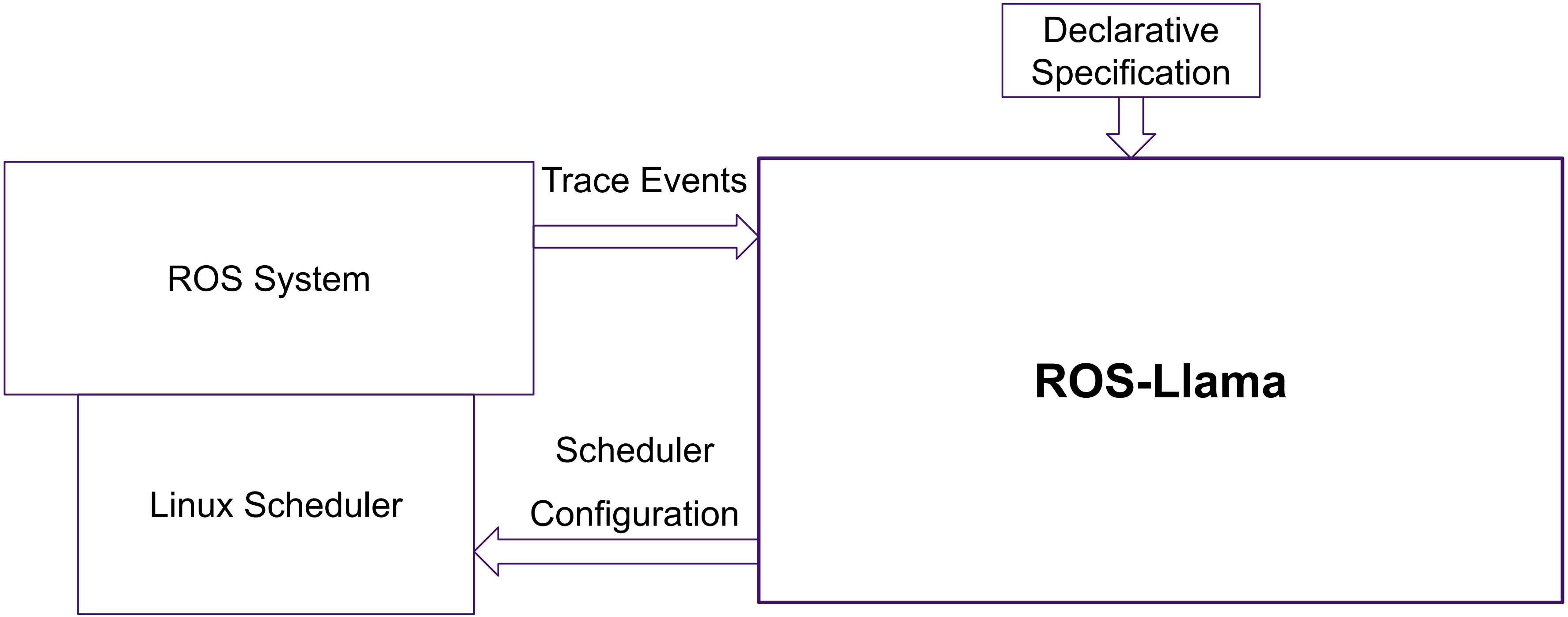
Requirement	Consequence
Ease of use • No need to know how included packages work • No need to know how callbacks are run • No need to know real-time theory	**Cannot ask the user** for the information required to bound response times
Below-worst-case provisioning	Must **degrade gracefully** in case of an overload
Runs on the **officially supported platforms**	Must use **mainline Linux** (+ PREEMPT_RT)

Tobias Blass

Outline

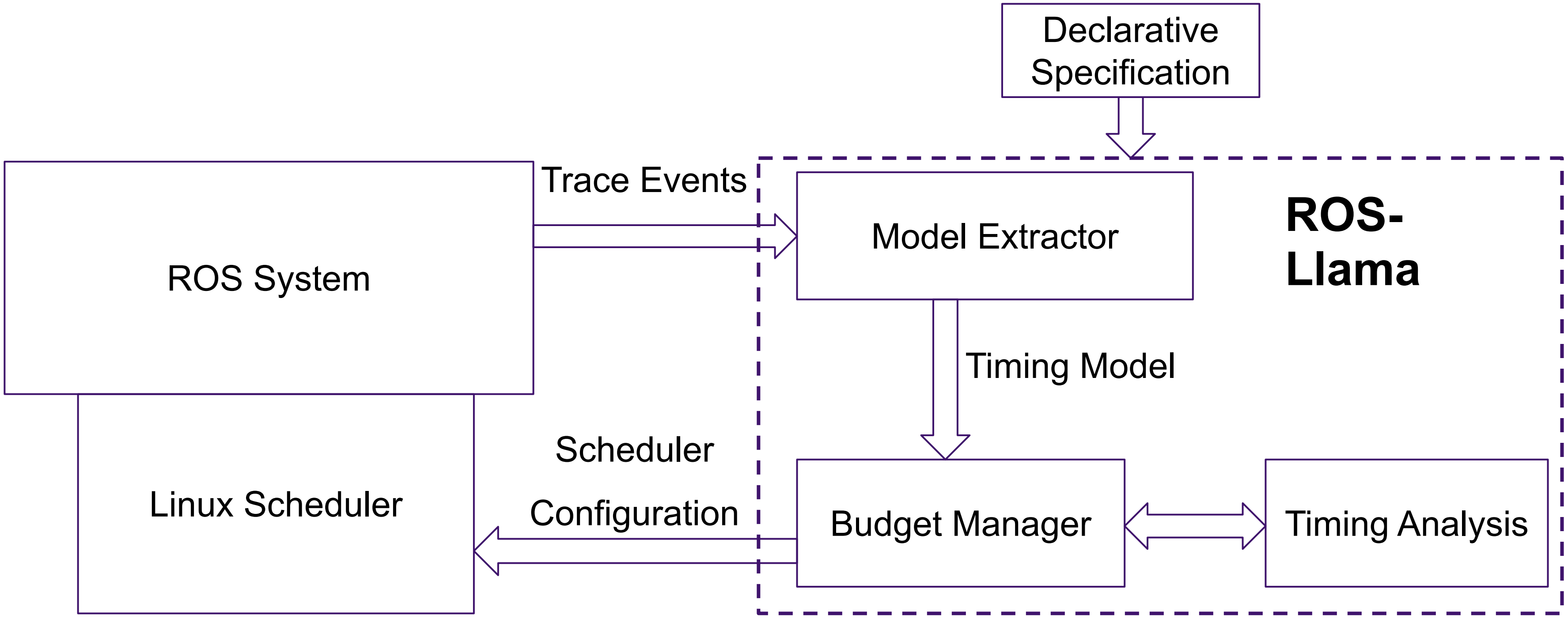
1. Background
2. Why is real-time theory difficult to apply to ROS?
3. The ROS Live Latency Manager (ROS-Llama)
4. Evaluation

ROS-Llama: Design



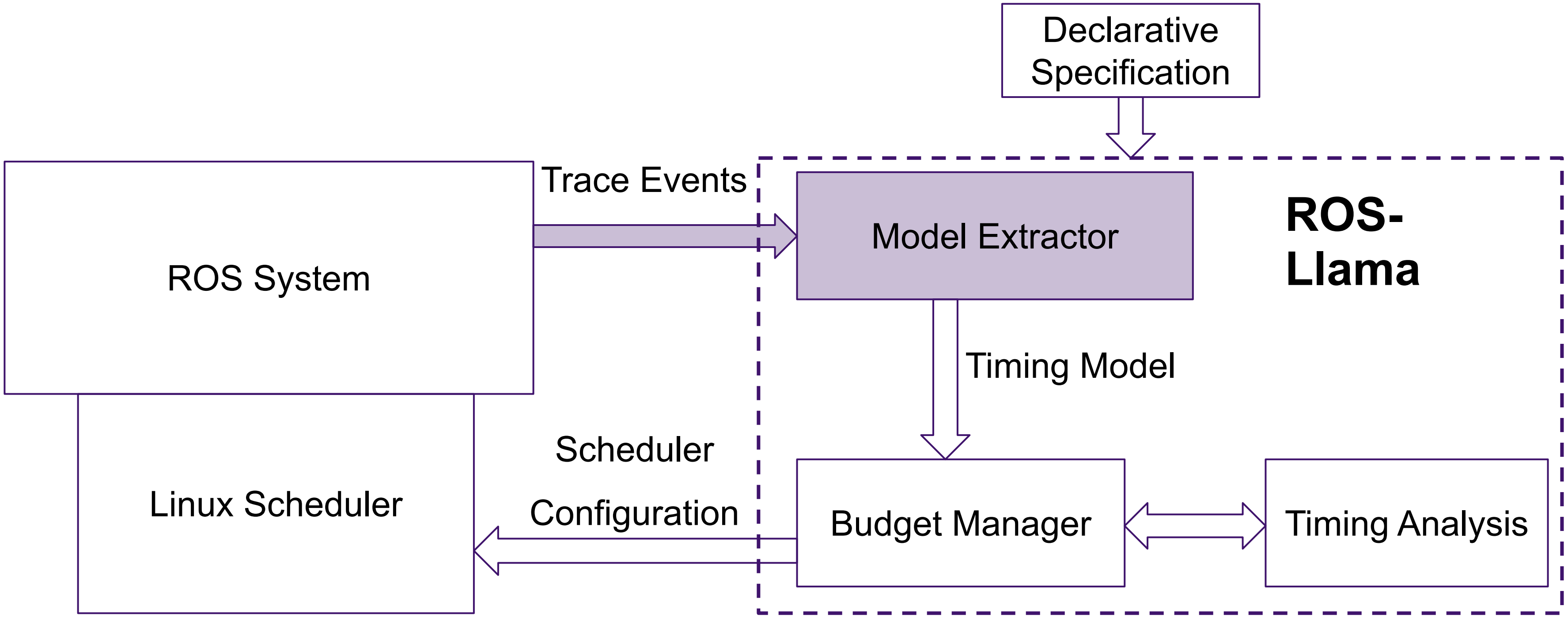
Tobias Blass

ROS-Llama: Design



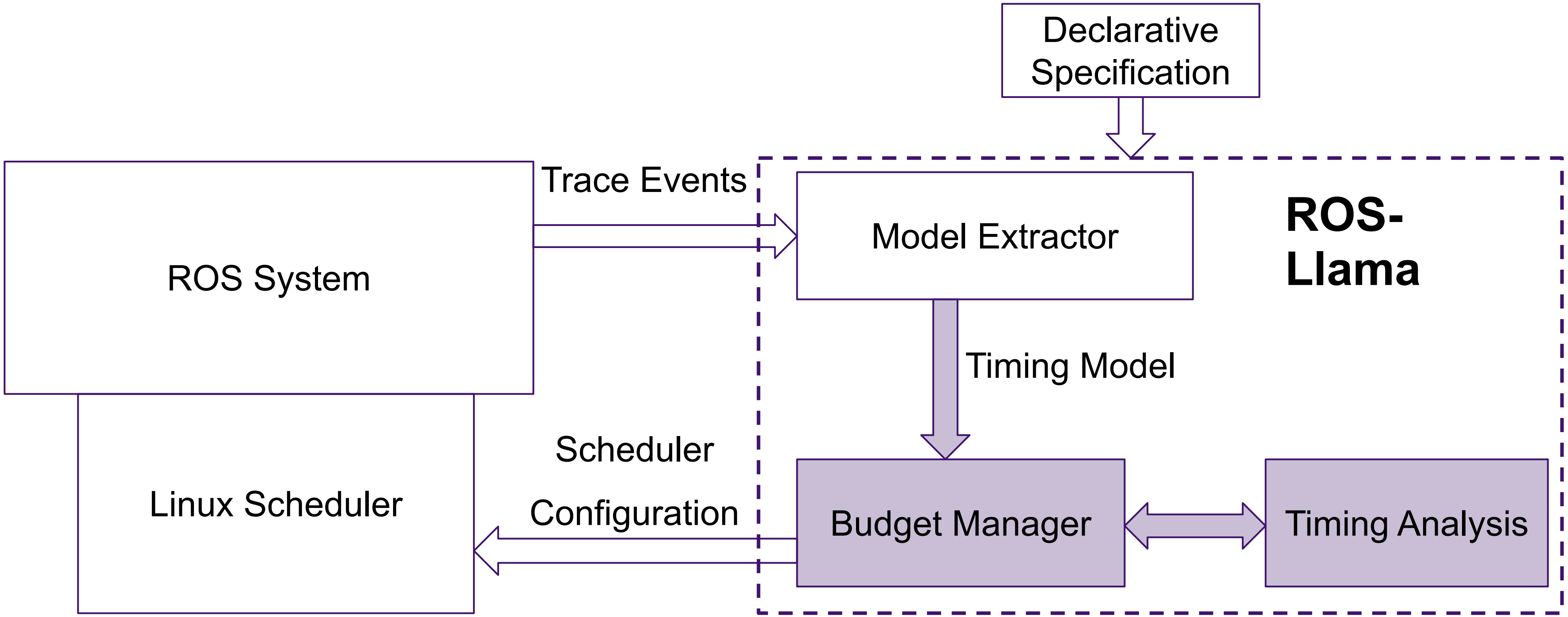
Tobias Blass

ROS-Llama: Design



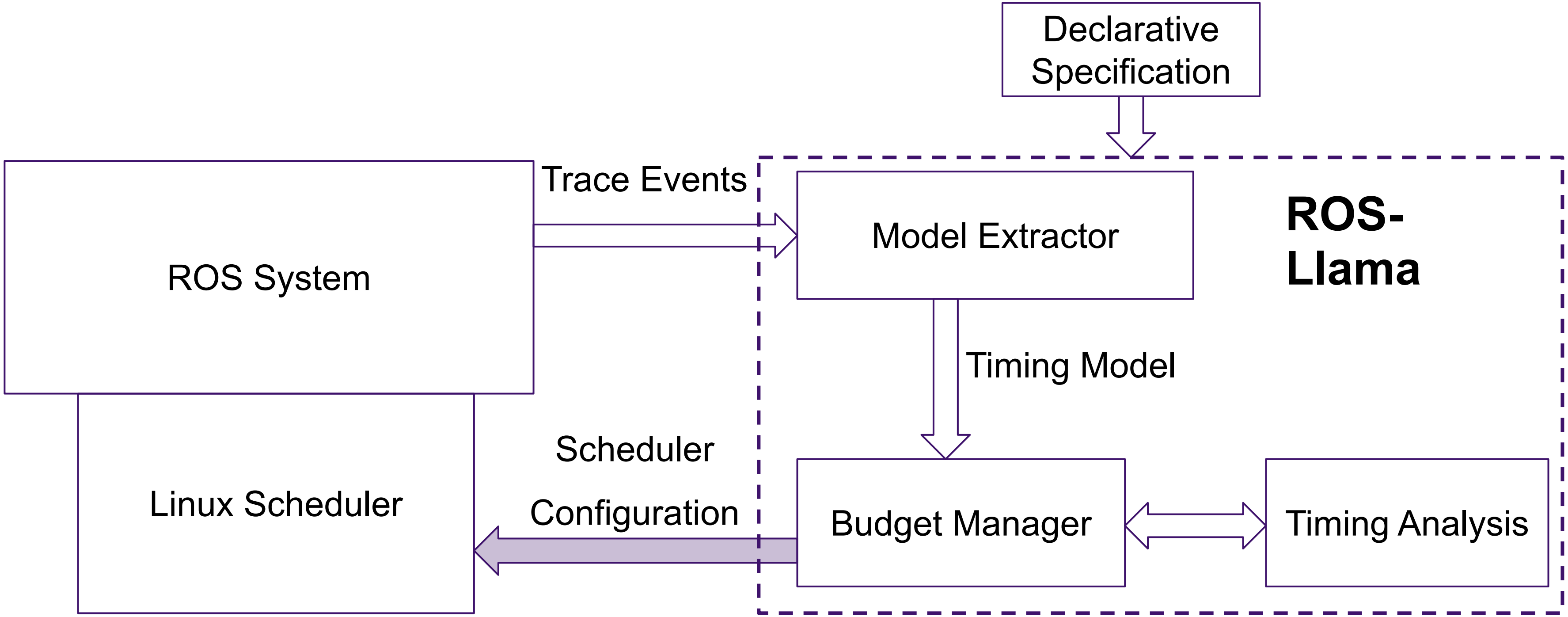
Tobias Blass

ROS-Llama: Design



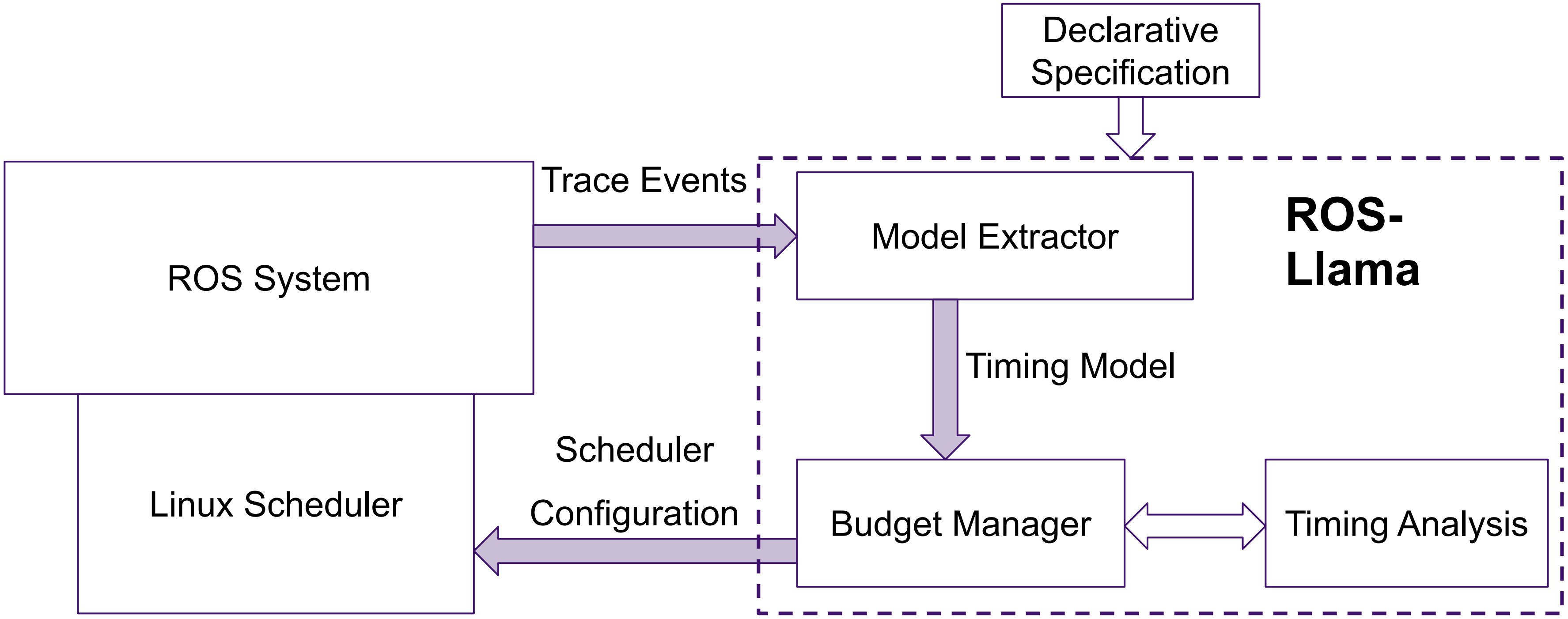
Tobias Blass

ROS-Llama: Design



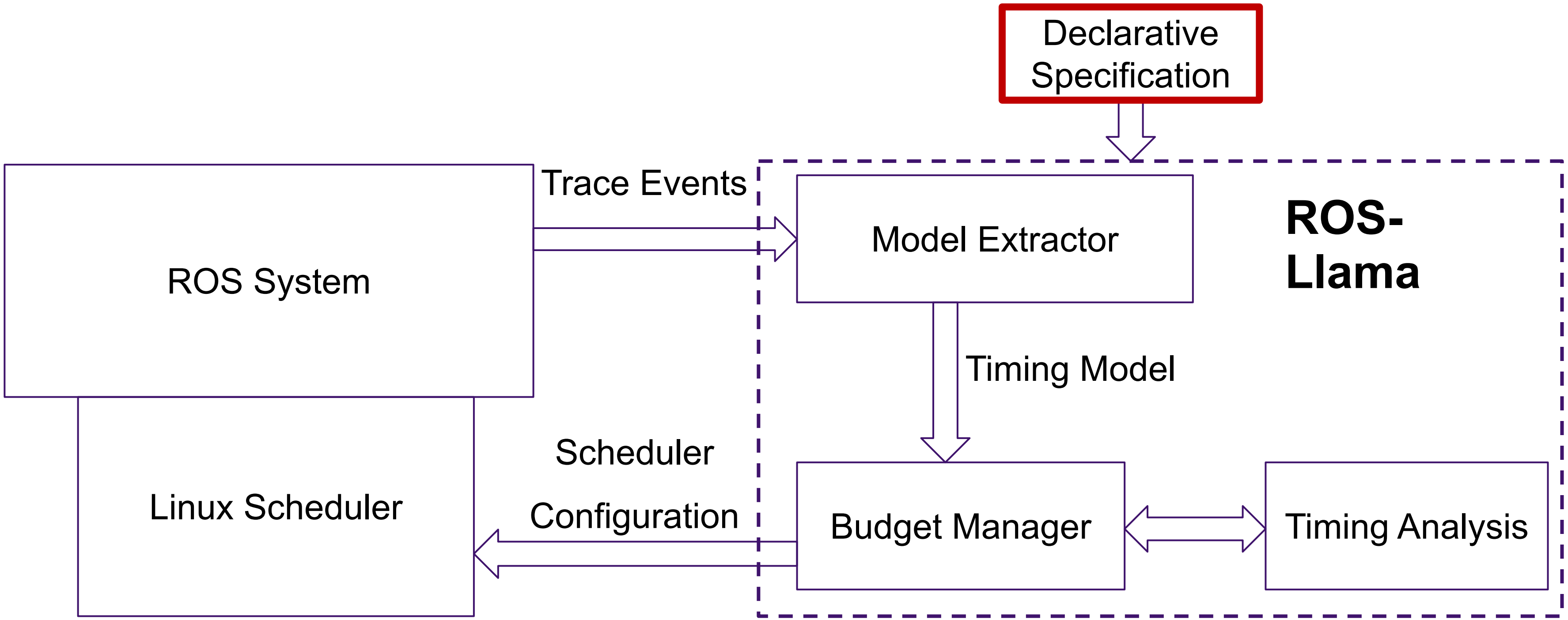
Tobias Blass

ROS-Llama: Design



Tobias Blass

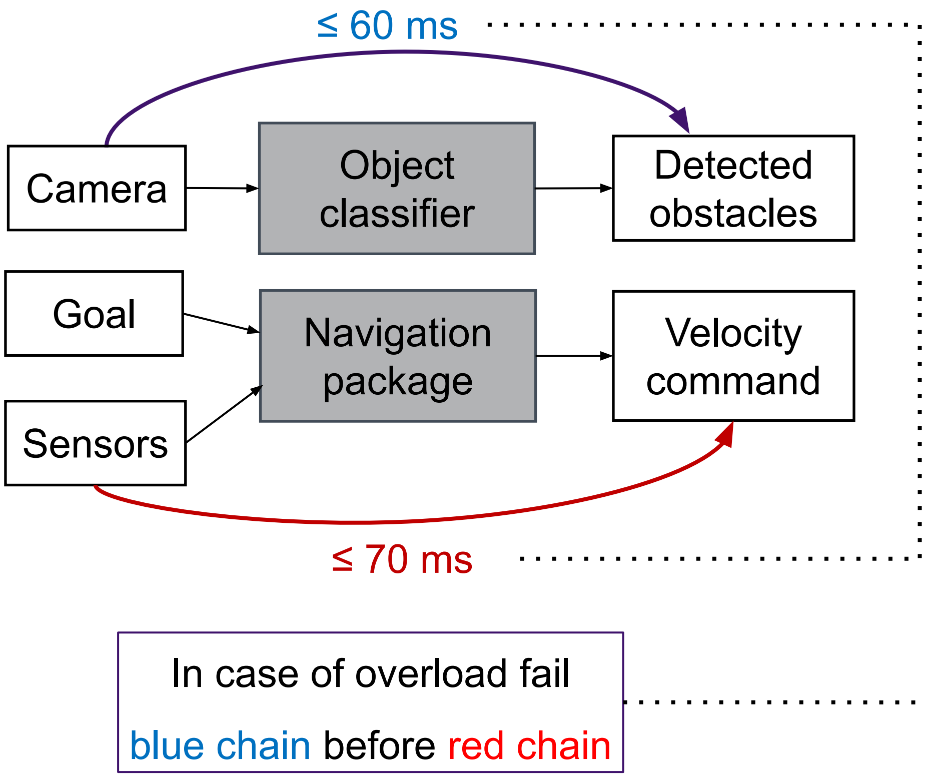
ROS-Llama: Design



Tobias Blass

BOSCH

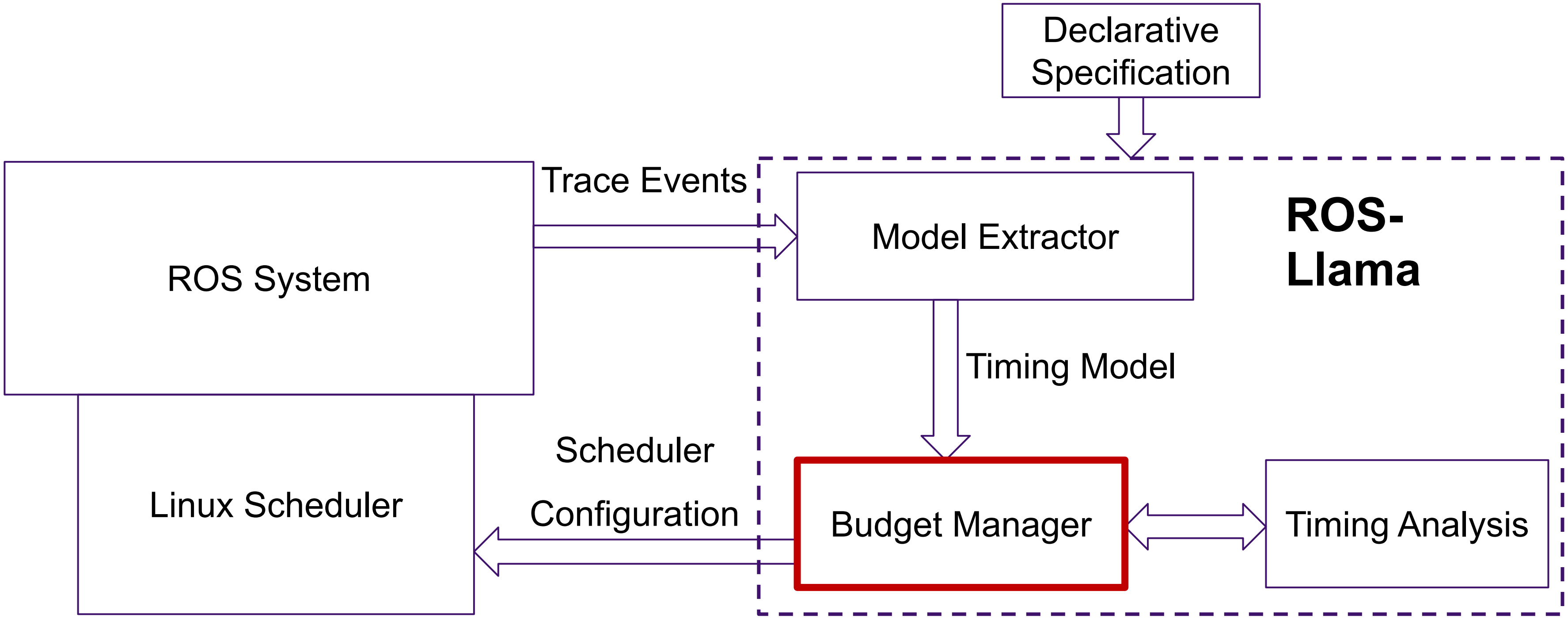
ROS-Llama: Specification



1. **End-to-end latency goals** for timing-critical processing chains
2. **Degradation order**

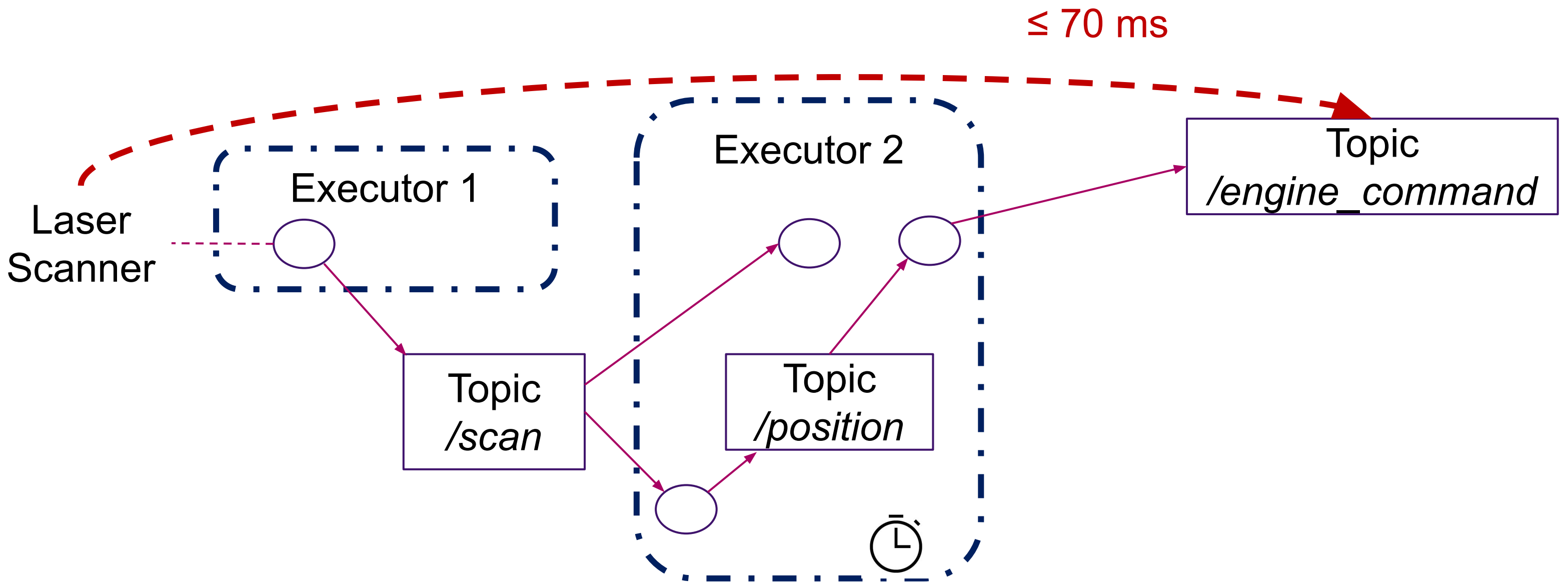
Tobias Blass

ROS-Llama: Design



Tobias Blass

The Budget Manager

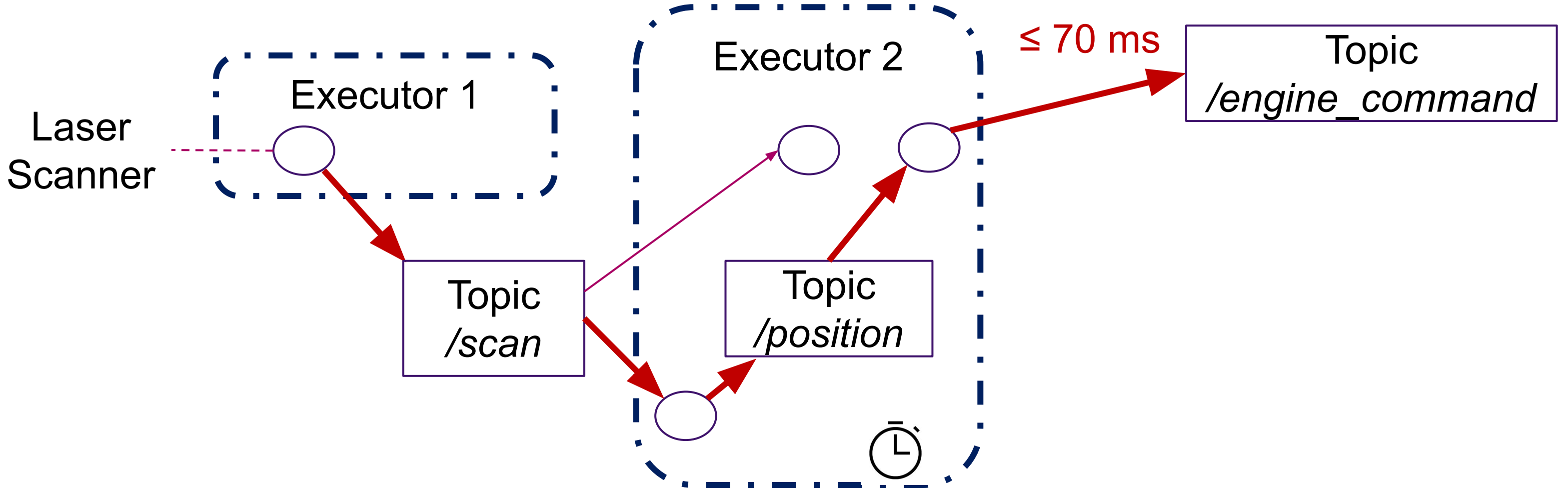


Step 1: Find relevant callbacks
Which callbacks are part of the designated processing chain?

Tobias Blass

BOSCH

The Budget Manager



Step 2: Scheduling Parameters
Which scheduling parameters for the executor threads guarantee the desired response time?

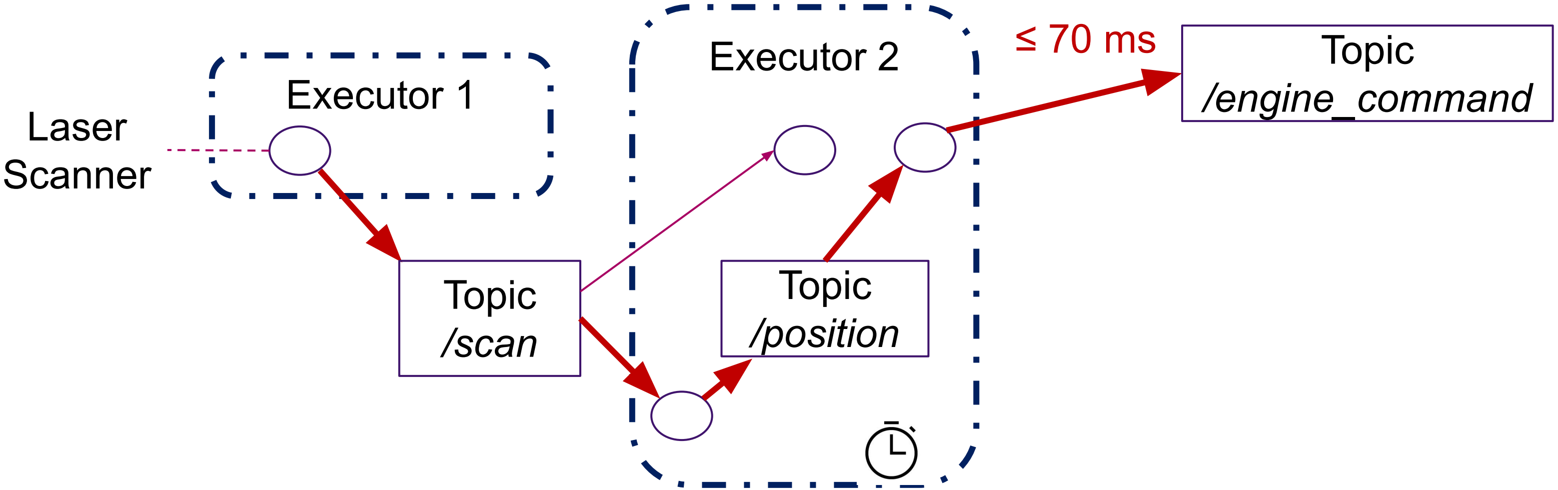
Tobias Blass

Background: The SCHED_DEADLINE Scheduler

- Linux's reservation-based scheduler
- Each thread has a **budget** and a **period**
- Threads are **guaranteed** to receive their budget in each period
- **Provides temporal isolation**

Tobias Blass

The Budget Manager



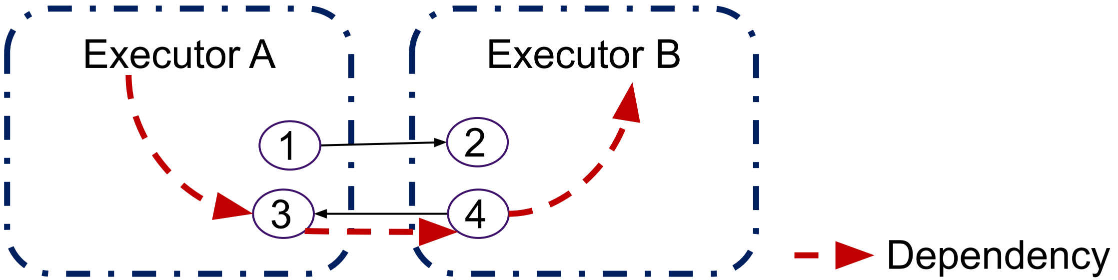
Step 2: Choose Budgets/Periods
Which scheduling parameters for the executor threads guarantee the desired response time?

Tobias Blass

How to Choose Budgets?

Example: Executor A

- Need to bound processor demand of callback 3
  - Depends on number of activations per period
    - Depends on response-time of callback 4
      - Depends on budget of executor B

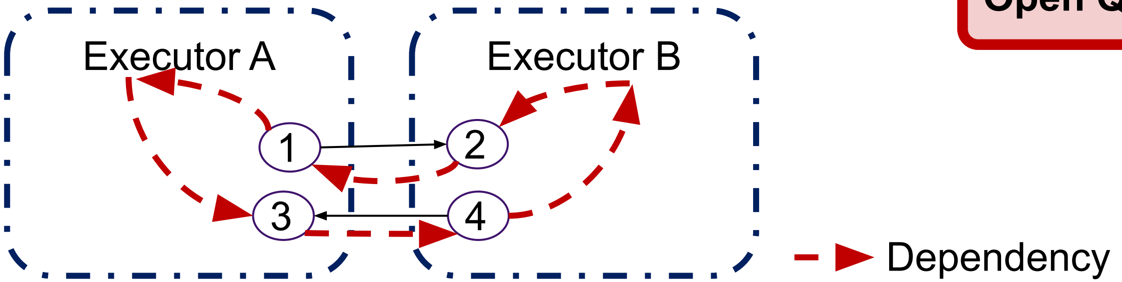


Tobias Blass

How to Choose Budgets?

Example: Executor B

- Need to bound processor demand of callback 2
  - Depends on number of activations per period
    - Depends on response-time of callback 1
      - Depends on budget of executor A



Complex global optimization problem

Solution: **greedy heuristic**

Open Question: Is there a better solution?

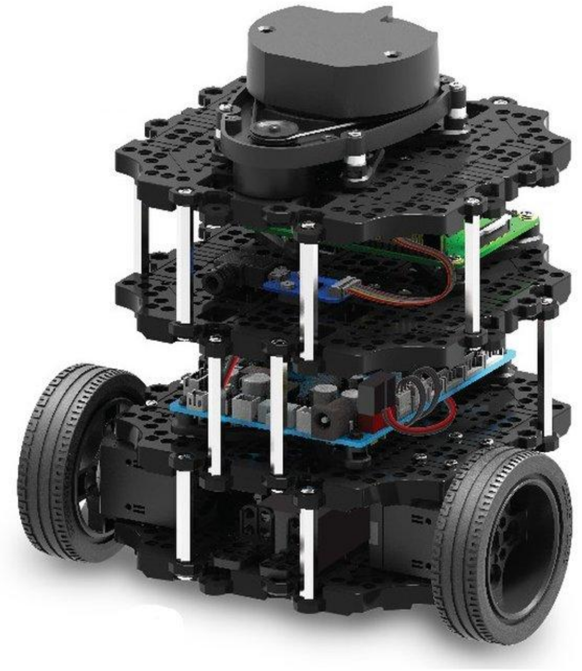
Tobias Blass

Outline

1. Background
2. Why is real-time theory difficult to apply to ROS?
3. The ROS Live Latency Manager (ROS-Llama)
4. Evaluation

Evaluation Setup

- **Turtlebot 3** “Burger” controlled by a **Raspberry Pi 4B**
  - Navigation 2 package
  - ROS 2 Object Analytics package (*Tracker*)
- Robot patrols between two points
- Tracker load **increases over time** in three phases



Tobias Blass

Evaluation Questions

1. Does ROS-Llama **fulfill** the navigation **latency goals**?

2. What if tracker load increases (**graceful degradation**)?

3. Would a **simpler solution** do just as well?

Default scheduler

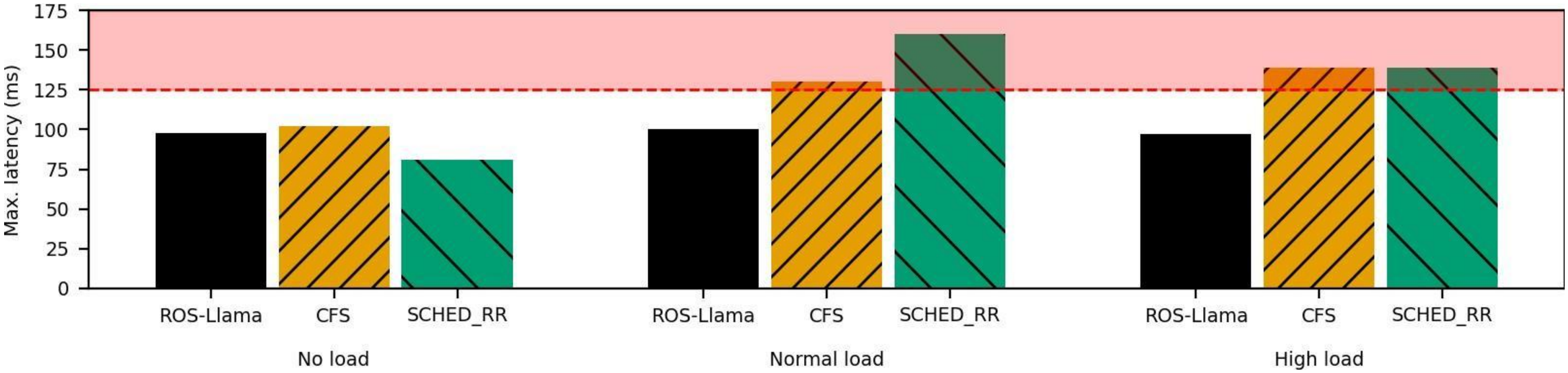
- **CFS** (fair-share scheduler)
  - Default Linux scheduler, no configuration required

RT scheduling without analysis

- **SCHED_RR** (fixed-priority scheduler)
  - Criticality-monotonic priorities for controlled degradation

Tobias Blass

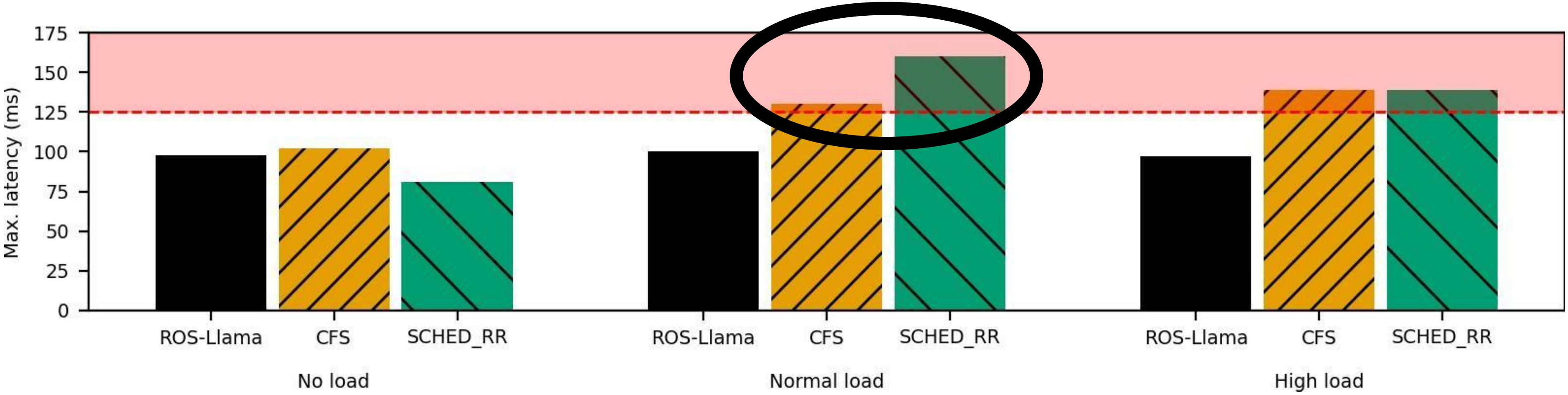
Evaluation Results: Pilot Chain



Only ROS-Llama manages latency well in all situations

Tobias Blass

Evaluation Results: Pilot Chain

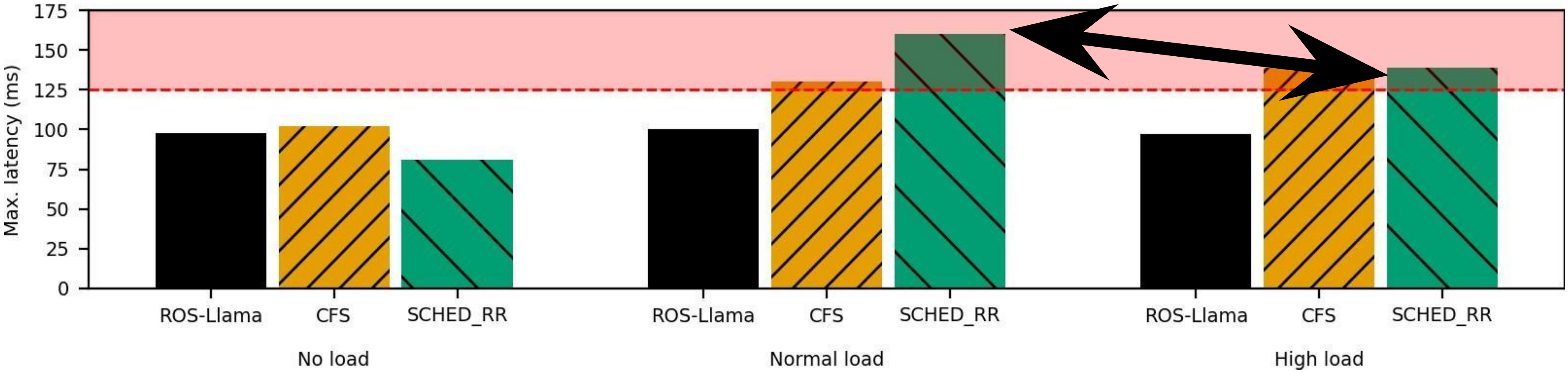


Only ROS-Llama manages latency well in all situations

General-purpose scheduling may work better than misconfigured real-time scheduling

Tobias Blass

Evaluation Results: Pilot Chain



Only ROS-Llama manages latency well in all situations

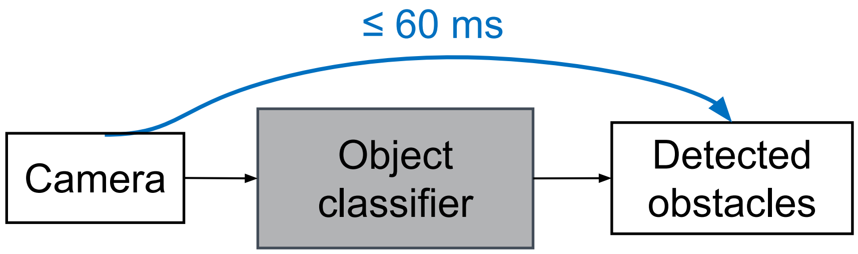
General-purpose scheduling may work **better** than misconfigured real-time scheduling

Impact of load is complex and **hard to predict without analysis**

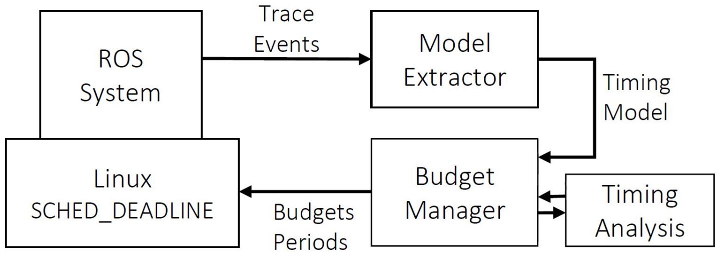
Tobias Blass

Summary

ROS-Llama: Easy, low-effort real-time scheduling for ROS



Simple declarative specification



Automatic scheduler configuration at runtime

In the paper: extensive discussion of **open problems**

- SCHED_DEADLINE limitations
- Linux I/O
- ROS Idiosyncrasies
- Complex Activations
- Stochastic Analysis
- DDS Analysis

Tobias Blass

Applying real-time theory in robotics is difficult

Rapid Development

ROS

Large open-source robotics framework & ecosystem

\+

Timing Correctness

Real-Time Systems Theory

Well-established theory

Abstracts details for simpler understanding and reuse

ROS-Llama

Use detailed description of the system to derive strong bounds

Tobias Blass