Reference Implementation of a novel Runtime Monitoring Architecture

Presented by: Humberto Carvalho

Supervised by:

Eduardo Tovar

Geoffrey Nelissen

isep Instituto Superior de Engenharia do Porto

CISTER
Research Center in
Real-Time & Embedded
Computing Systems

Our Agenda for Today

Who we are

Meet the men and women who are behind the curtain

Meet the gang

Those involved in the project

The problem

What we’re trying to fix

The Solution

The fix to the problem

The Achievements

(Finally)

Questions

Q&A

Who we are

Meet the men and women who are behind the curtain.

CISTER

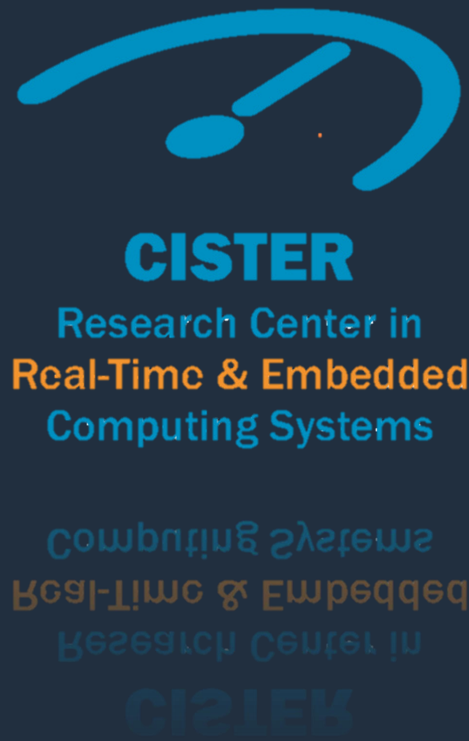
Who we are

CISTER (Research Centre in Real-Time and Embedded Computing Systems) is a research unit based at the Polytechnic Institute of Engineering in Porto (ISEP), Portugal.

CISTER is a key contributor to a number of research subjects:

- ✓ **Real-time communication networks and protocols**
- ✓ **Wireless sensor networks**
- ✓ **Cyber-physical applications**
- ✓ **Real-time programming paradigms and operating systems**
- ✓ **Distributed embedded systems**
- ✓ **Scheduling and schedulability analysis**

….



CISTER

Overview

Industry leading

One of the leading European research units in real time & embedded platforms.

People

Around 60 persons, half with PhDs and encompassing more than 20 nationalities.

CISTER
Research Center in
Real-Time & Embedded
Computing Systems

Excellency

Awarded the classification of excellency in the FCT (Foundation for Science and Technology) evaluations 2004 and 2007..

Key Partnerships

Leading European Projects include the support of AIRBUS, THALES, BMW, CRITICAL SOFTWARE and other industry leading companies.

Meet the gang

Those responsible for this project.

Meet the gang

Those involved in the project.

Eduardo Tovar

CISTER Director

Geoffrey Nelissen

Research Associate

Humberto Carvalho

Undergrad Student

The Problem

What we’re trying to fix

The Problem

Contextualization

Safety Critical Systems

Safety Critical Systems are composed of hardware and software responsable for some task whose **failure can have catasthropic consequences** where human life, infastructures are at risk.

The correctness of these systems depends not only on the **correct output but also the moment in time** that output is delivered. These systems are formally known as **real-time**.

Examples of these systems include:
Avionic Industry
Nuclear Industry
Automotive Industry

The Problem

System Certification

Application Correctness

Given the **high criticality** of these systems, it is crucial to formally guarantee they work correctly.

Certification

Certification Is a lengthy and expensive process in which the program **output and time requirements are verified to be correct** in all circumstances impacting the safety of the system. For a program to be used in a real time safety critical context, it must be certified.

Complexity Explosion

However in recent years the functionality required of computer systems for safety-critical real time applications has **increased exponentially**. Scaling computing capacity to meet this requirements became difficult, as traditional approaches to CPU performance stopped producing significant performance enhancements. **Forcing manufacters to come up with somewhat exotic optimizations that make the system ever less deterministic**, such as :

- Caching
- Data Prefetching
- Pipelining
- Branch Prediction
- Instruction level parallelism
- Instruction stream reorder for out-of-order execution
- Complex cache replacement policies
- **Multi-core**

The Problem

Certification Hurdles

Single Core Certification Complexity

Even for single core processors with most of their optimization mechanism disabled, it is **hard to prove that all the functional and timing requirements are respected.**

Multi Core Certification Complexity

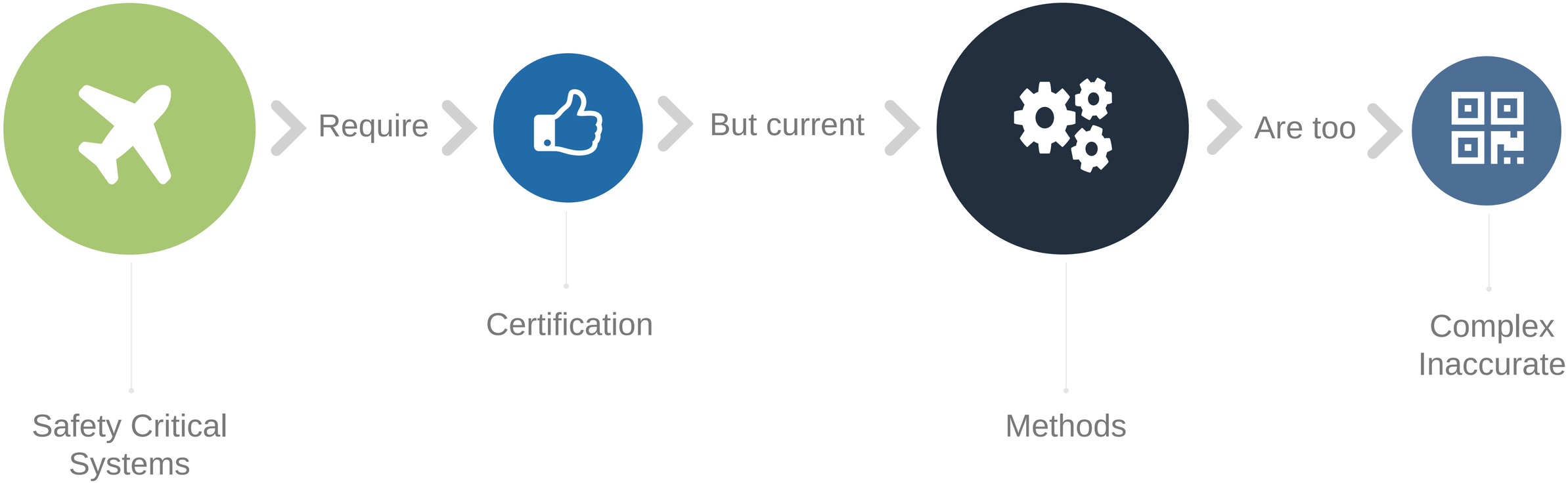
In multi core platforms, the exhaustive testing of all the situations encountered by a system becomes **infeasible to accuratly model, analyze and verify.**
One solution is to unrealistically always assume the worst case scenario, this analysis is sometimes so pessimistic that a task set which is schedulable in single core platforms is deemed unschedulable in multi core platforms.

Other problems

Other problems such as increased **software complexity** and the development practises used in the avionics, nuclear, railway and automotive industries, also difficult system verification. One example of these practises is **distributed development**, where independent modules developed by independent teams are later integrated into a final product.

The Problem

Wrap Up



The Solution

The fix to the problem

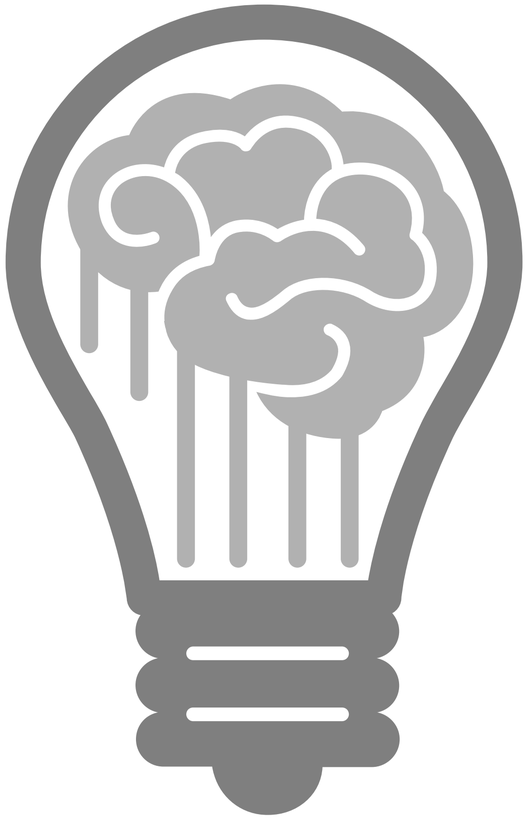
The Solution

Brainstorming

1 Calculating Probabilistic

Rather than worst case scenarios

A recent tendency is to calculate probabilistic rather than worst case scenarios for each given task-set.



2 Failure unlikely, but possible

In rare but nevertheless possible scenarios

However small the probabilities of failure might be, it is possible for a system to fail in rare but possible scenarios.

The Solution

Brainstorming - Problem Solution

1 Calculating Probabilistic

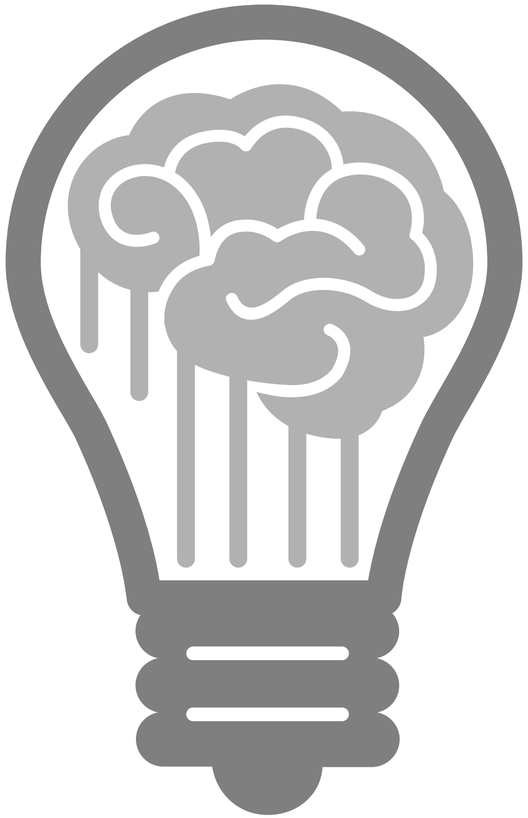
Rather than worst case scenarios

A recent tendency is to calculate probabilistic rather than worst case scenarios for each given task-set.

2 Failure unlikely, but possible

In rare but nevertheless possible scenarios

However small the probabilities of failure might be, it is possible for a system to fail in rare but possible scenarios.



Runtime verification 3

Online analysis

It becomes important to check at runtime the behavior of these system and ensure it is working properly within the given specifications.

Promising Solution 4

To help develop safety-critical systems

Runtime verification is a promising solution to help accelerate the development of safety-critical applications while upholding the standards associated with such systems.

The Solution

How does runtime verification work?

Monitors

Runtime Verification makes use of monitors: **applications which verify the specifications** which the system must respect at runtime.

They are able to **verify both functional properties and extra-functional properties by continually monitor the system**, and should any deviation from the specification arise, logs can be produced and **safe-guarding measures can be triggered in order to keep the system in a safe state**.

Functional Properties

Everything that relates to the produced result or execution ordering, such as: task A must execute before B, the sensor reading must be larger than 5.

Extra-functional Properties

Everything that does not relate to the result produced or order of execution, such as: a task A must complete within 10ms, power consumption must stay below 5W.

The Solution

Monitor Advantages

01 Efficiency

Monitors are an efficient solution to detect bugs and other deficiencies when an exhaustive verification is not feasible.

02 Guardians

They are guardians which can react to errors at runtime after the system has been deployed, increasing the safety of these systems during their continous operation.

The Solution

State of the art

Current Solutions

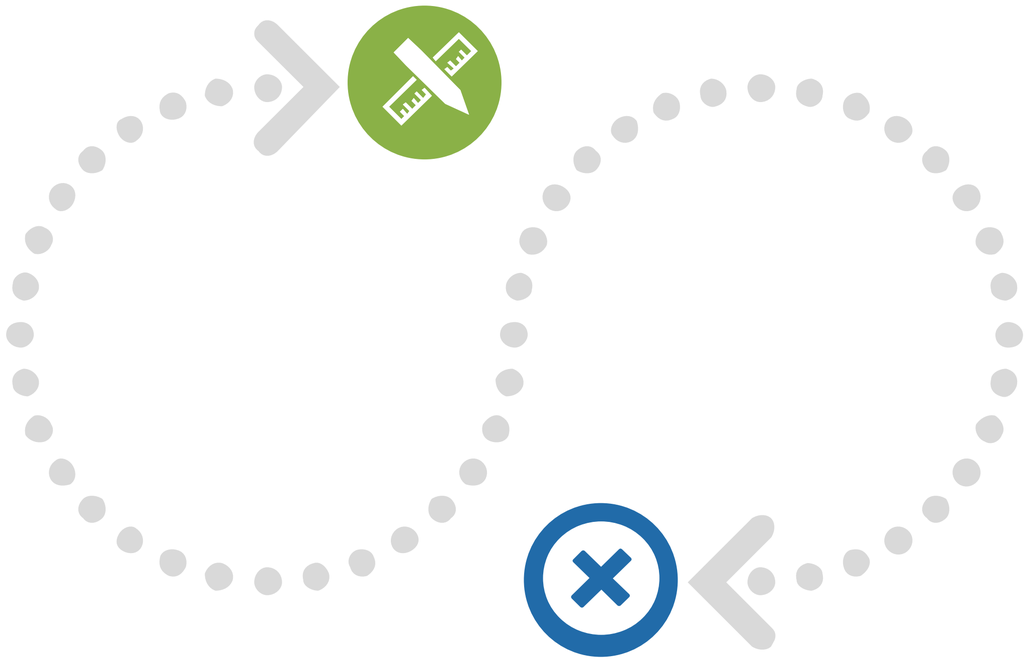
Considerable efforts in the area of runtime verifications are **focused on the specification language** and its translation in correct-by-constructions monitors.

The same cannot be said for the architecture in which these monitors operate, few of them address the problem of an efficient and safe architecture.

Problem

None of the solutions in the state of the art completely address the requirements associated with **high criticality applications.**

Without an **efficient and reliable infrastructure** to extract meaningful information, the concept of run-time verification becomes useless as its inputs and hence by extension its outputs cannot be trusted.



A novel state of the art architecture for runtime monitoring

- ✓ Suited for safety-critical applications
- ✓ Also suited for non-critical applications
- ✓ Addresses the requirements associated with high-criticality real time applications

CISTER
Research Center in
Real-Time & Embedded
Computing Systems

Developed by

- Geoffrey Nelissen
- David Pereira
- Luís Miguel Pinho

Independent and Composable Development

Allows for the distribution of the development of complex systems between different partners and subcontractors.

Time and Space Partitioning

Able of dividing system resources into partitions so that they are completely isolated from one another, and faults cannot propagate between them.

Simplicity

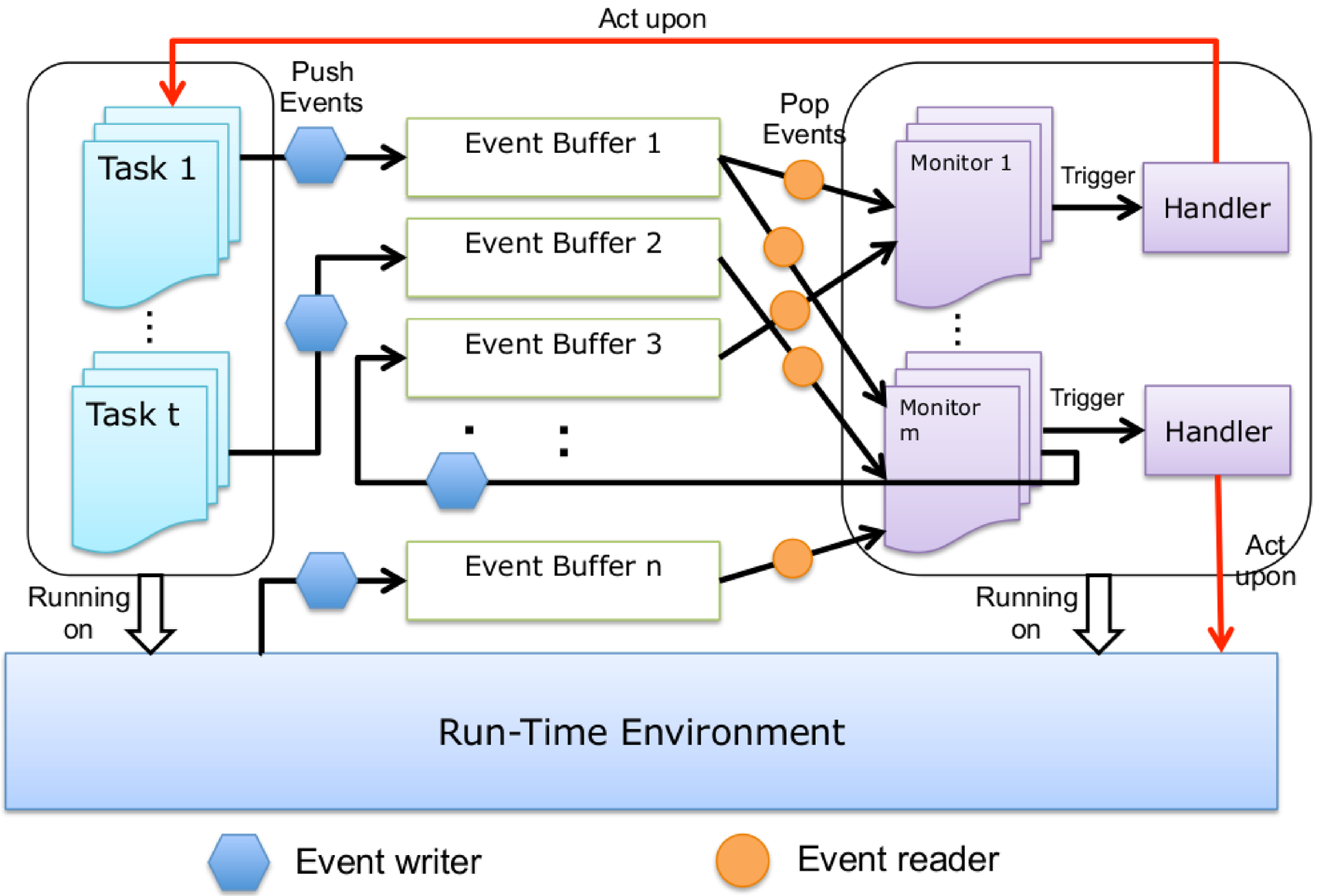
Simplicity is one of the main concerns for safety-critical systems since it eases the understanding of the system and the certification process.

Efficiency and Responsiveness

An efficient and responsive implementation ensures that information reaches the monitors as soon as possible. It is possible to monitor multiple tasks in parallel without any blocking time.

The Solution

The Architecture (brief overview)



CISTER
Research Center in
Real-Time & Embedded
Computing Systems

The Project
(finally)

The Project

What this is all about

Develop Architecture Data Structures and Algorithms

Although the architecture is well defined, its supporting data structures and related algorithms are not

Stand Alone C++ Implementation

POSIX compatible stand alone C++ Library

Linux Implementation

Implement the architecture as a loadable Linux kernel module

The Project

The Event Buffer Data Structure

Event based

Each dataset pushed onto the data structure is characterized by a timestamp.

Writer Priority

The write operation always occurs in constant time, minimizing impact on the monitored applications and easing the certification effort.

Wait-free

Writes, and in some conditions reads, are always wait-free: there is an upper bound on the number of instructions before each operation succeeds.

Highly Scalable

Readers are completely unaware of other readers, only aware of a single writer. Thus readers do not impact the performance of other readers.

Role Segregation

Writing is performed by Event Writers while reading is performed by Event Readers.

Data Deduplication

Data is never deleted from the event buffer, only overwritten, pushed events are read by all event readers as long as they're scheduled frequently enough.

Responsive

The wait-free property of the data structure allows for fast, predictable and in some cases deterministic operations on the data structure.

Highly Concurrent

One writer and many readers can operate from the same buffer at once.

The Project

Project Completion Status

Project Completion Status

100% The project has been completed succesfully.

Documented

The data structures, related algorithms and respective implementations have been thorougly analized and documented.

Tested

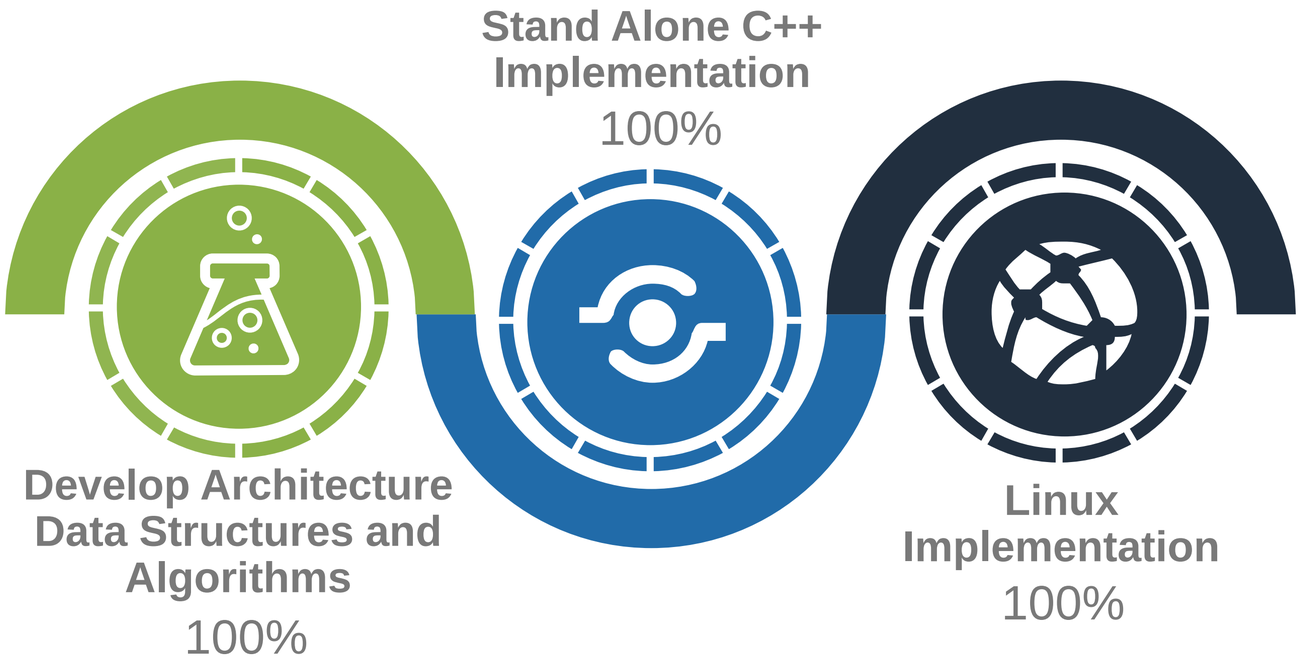
All implementations have been carefuly tested in both single threaded and multi threaded scenarios.

Used Internally

The project is currently being used in other projects inside CISTER.

Aproved

The project has been approved by the projects leaders.



The Project

Related Projects

Automatic Monitor Code Generation

The automatic generation of Monitors to a programming language is being developed in CISTER.

Monitor Specification Language

Sangeeth Kochanthara is designing a new monitor specification language suited to real time safety critical systems that is easy to use.

Implementation on an ARINC 653 Avionics compliant Operating System

An original goal of the project, which was postponed because of difficulties in obtaining such an OS.

The Project

Next Steps

Benchmark

Benchmark versus other solutions

Open Source

Make the project available to all.

Publish

Publish in a conference such as RTSS or RTAS.

Maintain

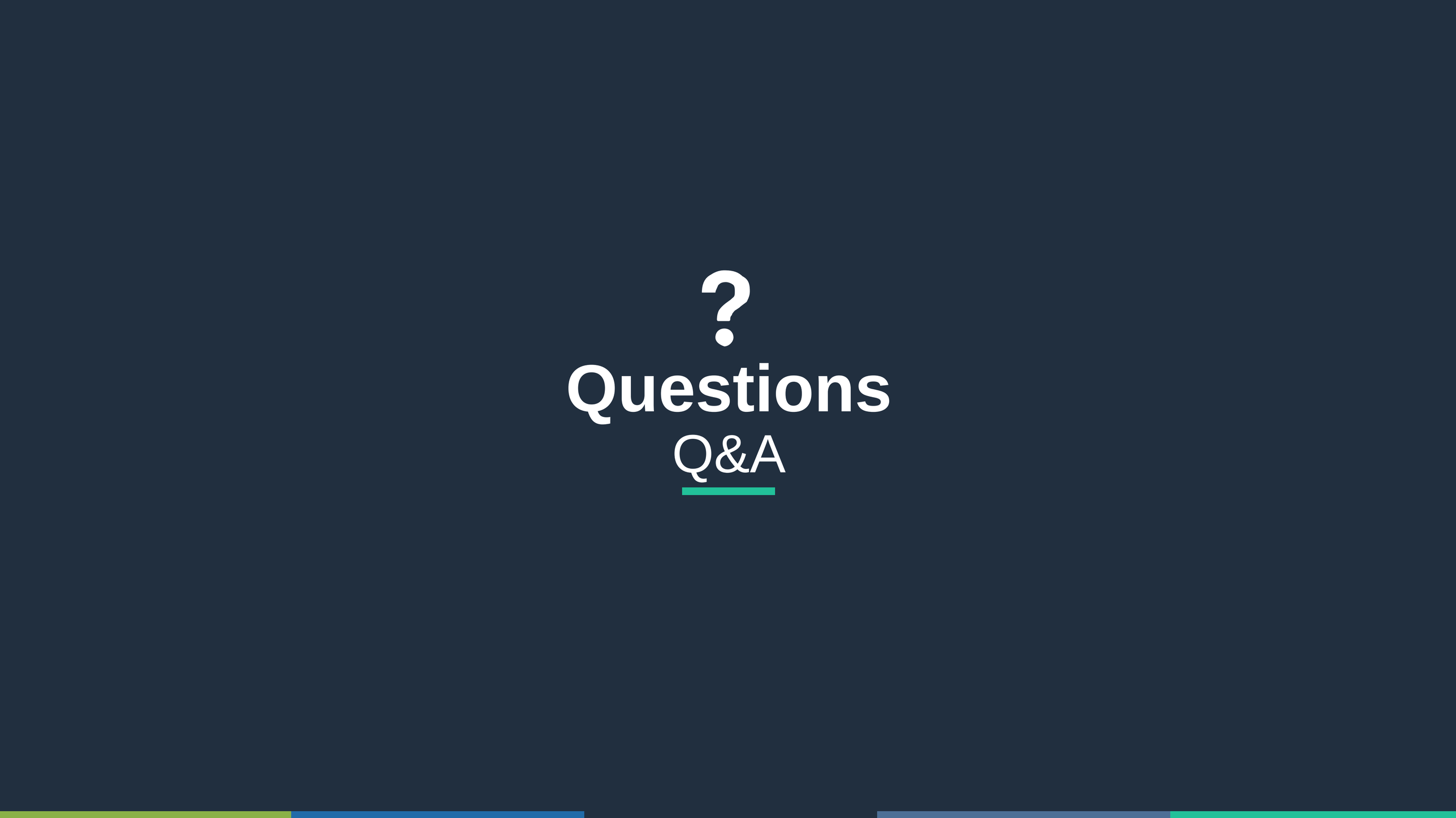
Maintain compatibility with newer kernels.

Improve

There is always room for improvement!

Move On

Move on to other critical problems at CISTER.


?
Questions
Q&A