The Polling Effect on the Schedulability of Distributed Real-Time Systems

Héctor Pérez (perezh@unican.es)
J. Javier Gutiérrez (gutierjj@unican.es)
Michael González Harbour (mgh@unican.es)
J. Carlos Palencia (palencij@unican.es)

Grupo de Ingeniería Software y Tiempo Real
Universidad de Cantabria
www.istr.unican.es

Financiado por el Gobierno de España y los fondos FEDER
(TIN2011-28567-C03-02/HI-PARTES y TIN2014-56158-C4-2-P (M2C2))

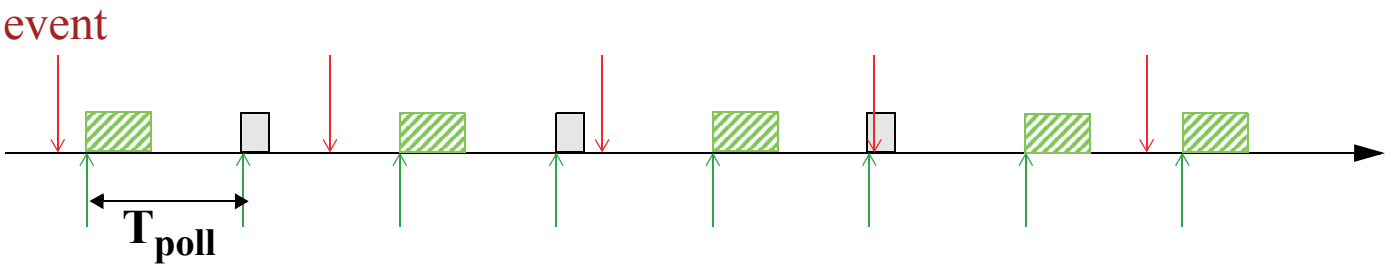
21st Int. Conf. on Reliable Software Technologies (Ada-Europe) 2016

Introduction

Our definition

A task that periodically polls for the arrival of its triggering event.

Thus, it executes its regular code only when detecting that the event had arrived



Motivation (1/2)

UNIVERSIDAD DE CANTABRIA

Polling is still relevant in today's distributed real-time systems

- systems able to deal with the latest available data
  - decouples different subsystems
  - simplifies schedulability analysis

Support included in software standards

- Data Distribution Service for real-time systems
- ARINC-653 for avionics

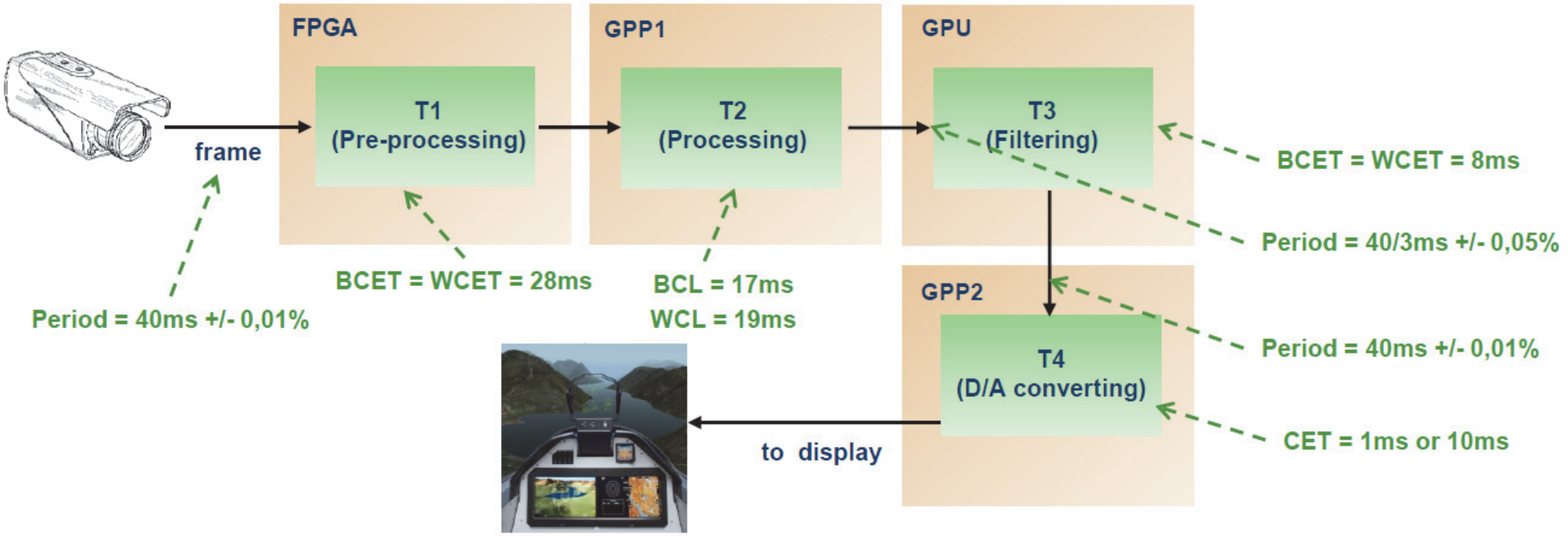
Some examples

- Traffic intersection control system (*Advanced Technologies* Inc)
- Lego EV3 Mindstorms with ev3dev Linux kernel

Motivation (2/2)

FMTV 2015 challenge from industrial case studies

- Aerial video tracking system



- https://waters2015.inria.fr/files/2014/11/FMTV-2015-Challenge.pdf

Event-driven applications

UNIVERSIDAD DE CANTABRIA

Features of *event-driven* distributed systems

- no loss of events in the system
- support included in software standards
- supported by holistic schedulability analysis and optimization techniques

Polling can be applied to event-driven distributed systems

Event-driven applications

UNIVERSIDAD DE CANTABRIA

Features of *event-driven* distributed systems

- no loss of events in the system
- support included in software standards
- supported by holistic schedulability analysis and optimization techniques

Polling can be applied to event-driven distributed systems

What are the costs in terms of schedulability?

Objectives

UNIVERSIDAD DE CANTABRIA

1. Propose a real-time model for the analysis of polling tasks
   - enables engineers to assess the effects of polling in their systems
   - integration in the MAST toolsuite
2. Quantify the effect of polling in event-driven applications
   - Using different scenarios for a representative example
   - *Analysis 1*: Response time analysis
     - obtain the worst-case response times using schedulability analysis techniques and the proposed real-time model for polling tasks
   - *Analysis 2*: Performance analysis
     - obtain the average response times in a real platform

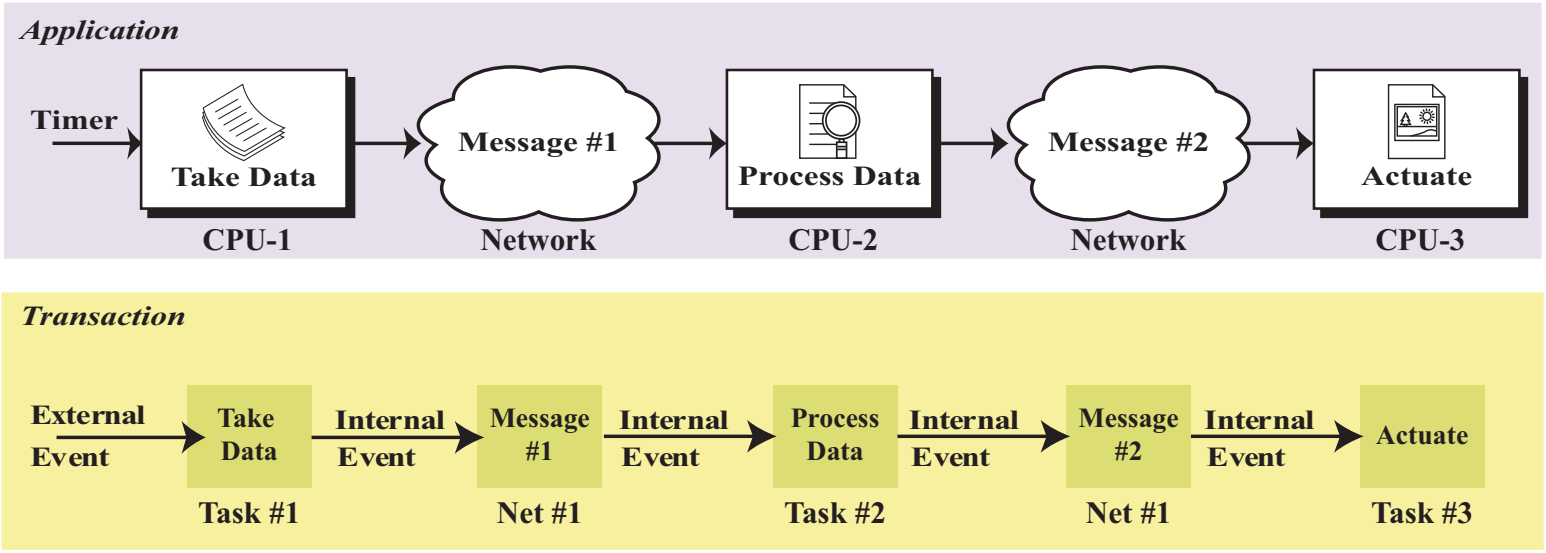
Objectives

1. Propose a real-time model for the analysis of polling tasks
   - enables engineers to assess the effects of polling in their systems
   - integration in the MAST toolsuite
2. Quantify the effect of polling in event-driven applications
   - Using different scenarios for a representative example
   - *Analysis 1*: Response time analysis
     - obtain the worst-case response times using schedulability analysis techniques and the proposed real-time model for polling tasks
   - *Analysis 2*: Performance analysis
     - obtain the average response times in a real platform

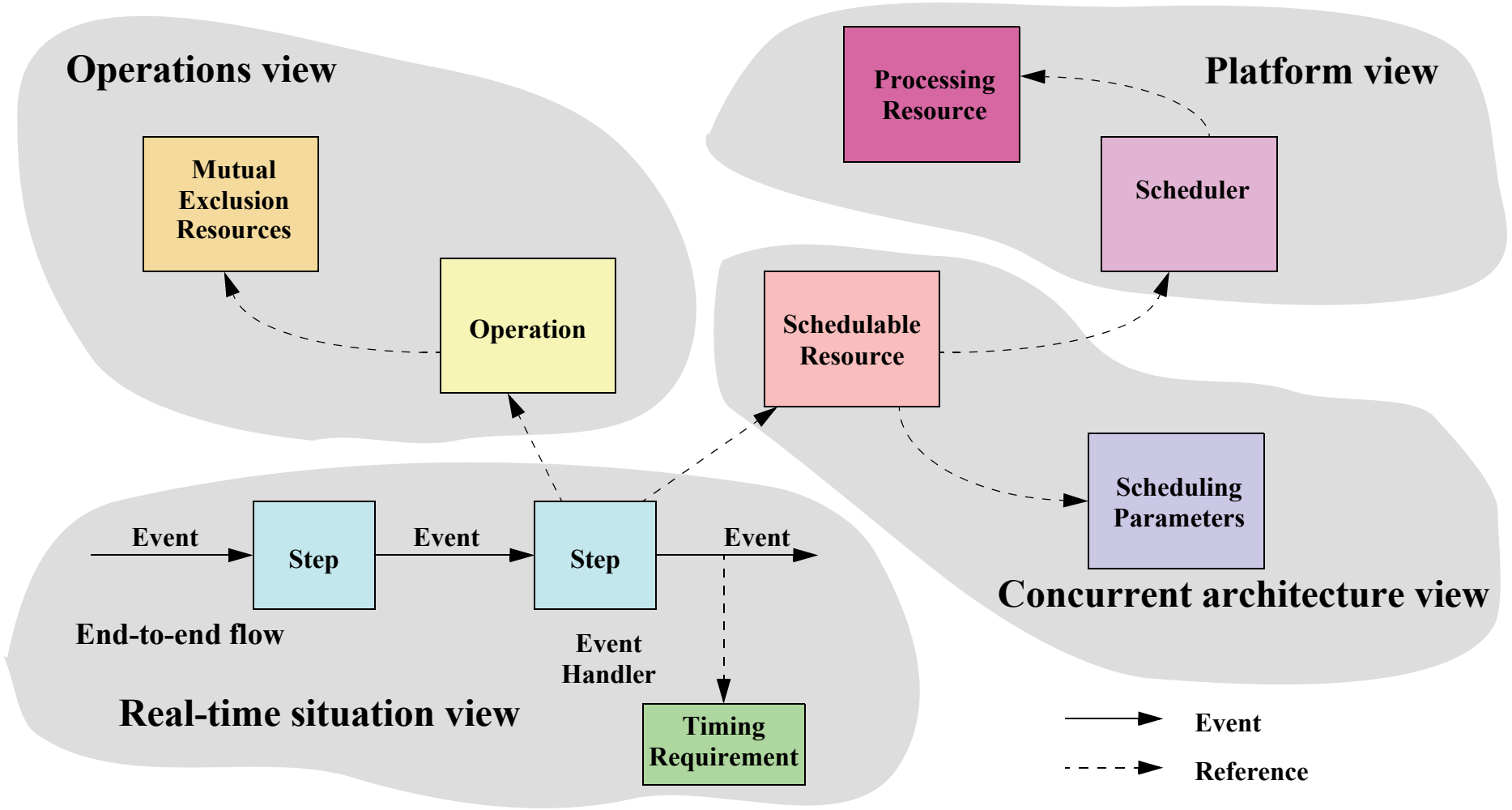
System model

End-to-end flow model from OMG MARTE modeling standard

- Used in schedulability analysis of event-driven applications
- Two schedulable entities: *tasks* for the processors, *messages* for the networks



MAST Overview



Polling in MAST

UNIVERSIDAD DE CANTABRIA

Scheduling mechanism by which there is a periodic task that polls for the arrival of its input event

- Represented as a *Scheduling Parameters* object
  - contains information on the scheduling policy and parameters used
- Common parameters:
  - *Priority*: the scheduling priority
  - *Preassigned:* assignment of the priority using optimization tools
- Special parameters:
  - *Polling Period*: period of the polling task
  - *Polling Overhead*: overhead of the polling task

Equivalent model for analysis in MAST

UNIVERSIDAD DE CANTABRIA

Polling tasks are modeled as two separate periodic tasks with jitter

- representing the polling task and its overhead
- regular schedulability analysis techniques can be applied

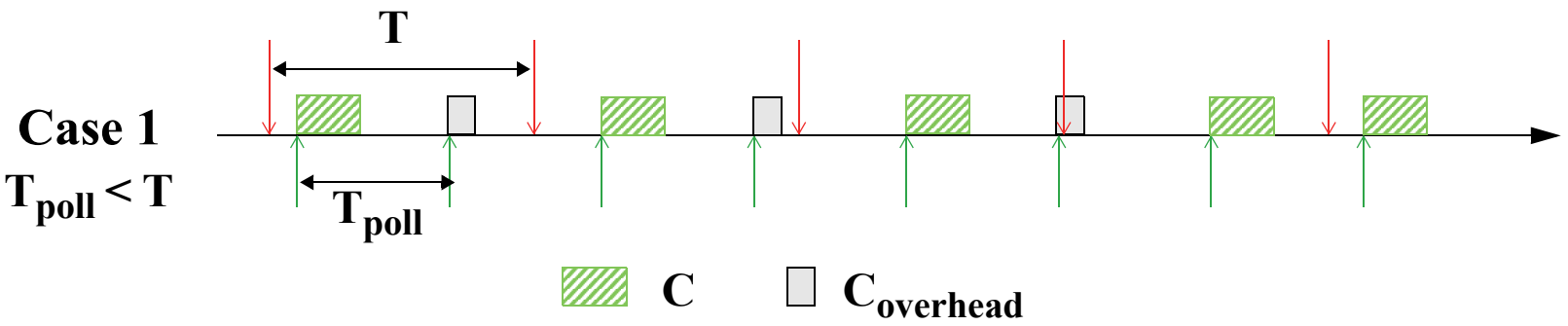
Response time analysis is determined by:

- the analysis of the own polling task
- the effect of polling on the analysis of lower priority tasks
- the overhead effects

Three different cases are identified according the relation between

- T, external event activation period
- T_{poll}, polling period

Polling and the response time analysis: Case 1



Analysis of the own polling task

- Regular analysis plus the T_{poll} delay

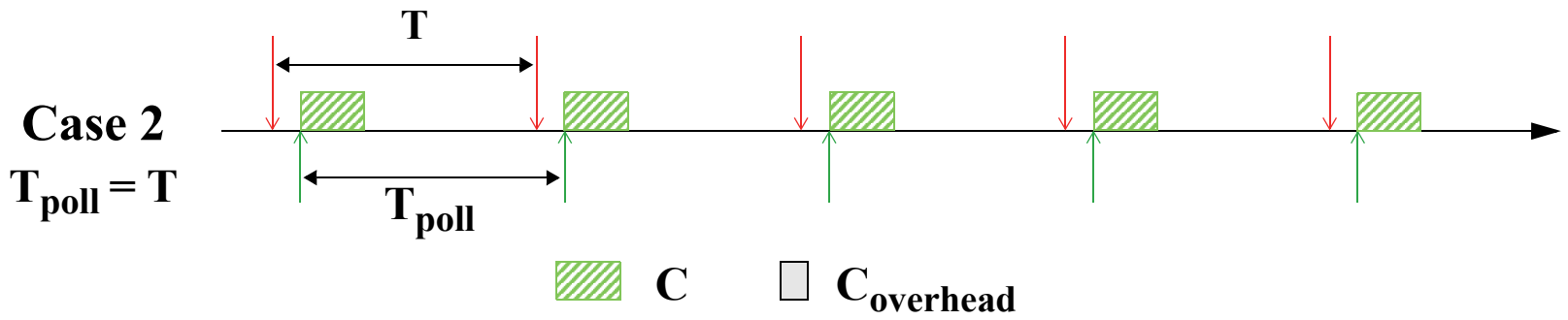
Analysis of lower priority tasks

- Equivalent periodic task with period = T and additional jitter of T_{poll}

Analysis of the overhead effects

- New independent periodic task with T_{over}, C_{over} and J_{over}

Polling and the response time analysis: Case 2



Analysis of the own polling task

- Regular analysis plus the T_{poll} delay

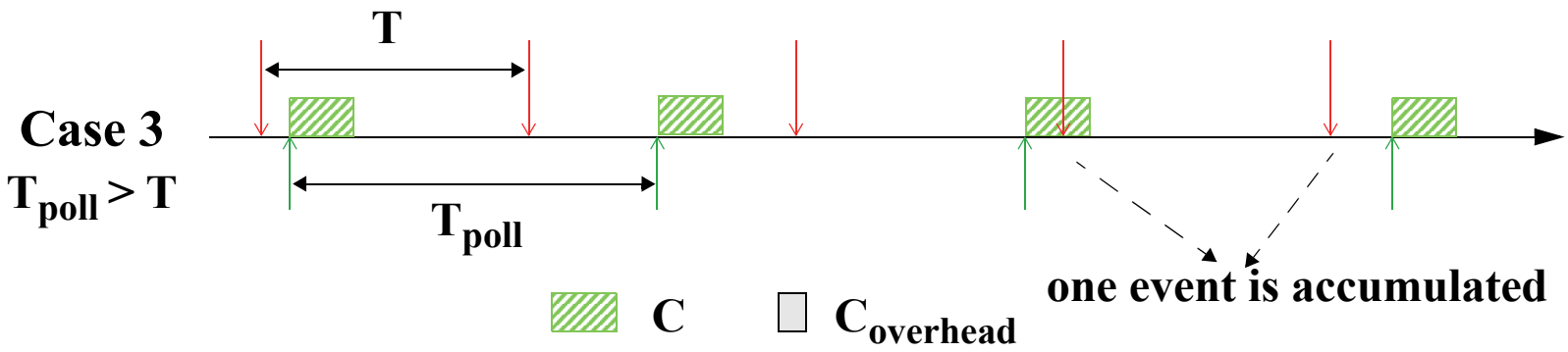
Analysis of lower priority tasks

- Equivalent periodic task with period = T (or T_{poll})

Analysis of the overhead effects

- No overhead caused by polling

Polling and the response time analysis: Case 3



Analysis of the own polling task

- Unbounded response time

Analysis of lower priority tasks

- Equivalent periodic task with period = T_{poll}

Analysis of the overhead effects

- No overhead caused by polling

Objectives

UNIVERSIDAD DE CANTABRIA

1. Propose a real-time model for the analysis of polling tasks
   - enables engineers to assess the effects of polling in their systems
   - integration in the MAST toolsuite
2. Quantify the effect of polling in event-driven applications
   - Using different scenarios for a representative example
   - *Analysis 1*: Response time analysis
     - obtain the worst-case response times using schedulability analysis techniques and the proposed real-time model for polling tasks
   - *Analysis 2*: Performance analysis
     - obtain the average response times in a real platform

Reference example (1/2)

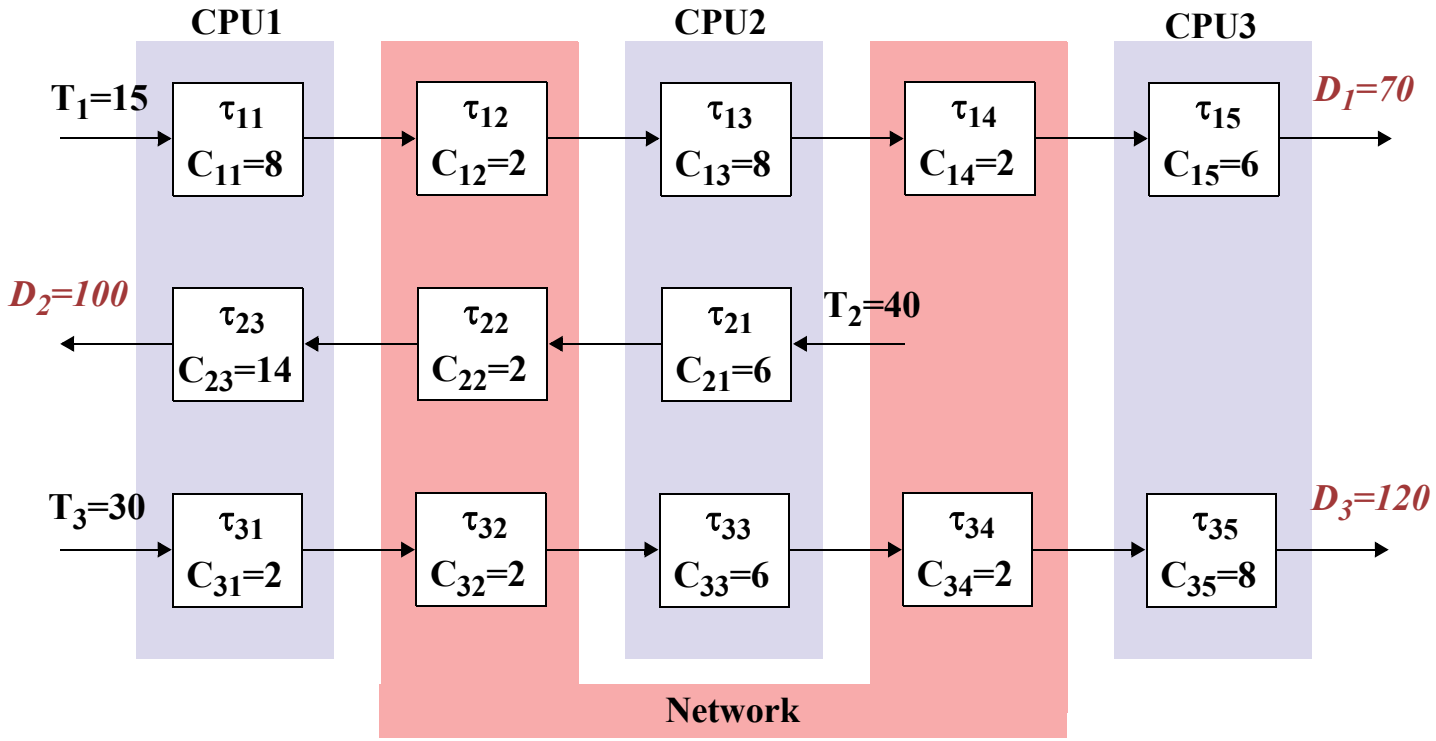
In *"Optimizing End-to-End Latencies by Adaptation of the Activation Events in Distributed Automotive Systems"* **(RTAS, 2007)**

- **illustrates the drawbacks of two activation models common in the automotive industry**
  - ***periodic*** **activation: tasks and messages are executed using polling**
  - ***event-driven*** **activation: tasks and messages are blocked until the completion of the previous step**
- **alternative response time analysis for *periodic* activation**
  - **valid for $T_{poll} = T$**
- **holistic analysis for *event-driven* activation**
- **proposes mixing *periodic* and *event-driven* activations to optimize system schedulability**
  - **tasks and messages priorities are fixed in advance**

Reference example (2/2)

Original example uses *Deadline Monotonic* priorities

- according to the E2E deadline
- assigned in decreasing order from each external event



Scenarios for the reference example

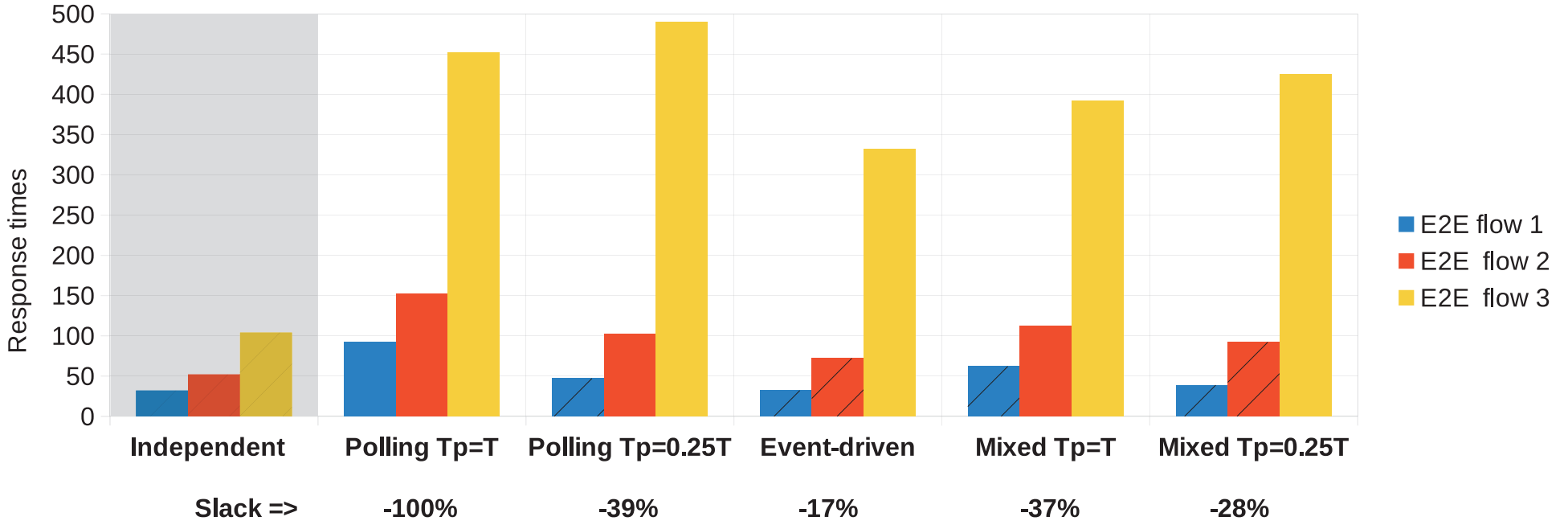
UNIVERSIDAD DE CANTABRIA

- *Scenario 1*: independent task model
  - tasks and messages are executed periodically as if they were independent
  - the notion of processing a particular event is lost
  - used as a reference scenario
- *Scenario 2*: polling model
  - tasks and messages are executed using polling
  - periodic activation model in the reference example
- *Scenario 3*: event-driven model
  - tasks and messages are blocked until the completion of the previous step
- *Scenario 4*: mixed model
  - only tasks are executed using polling

Response time analysis using *Offset_Based_Slanted* technique

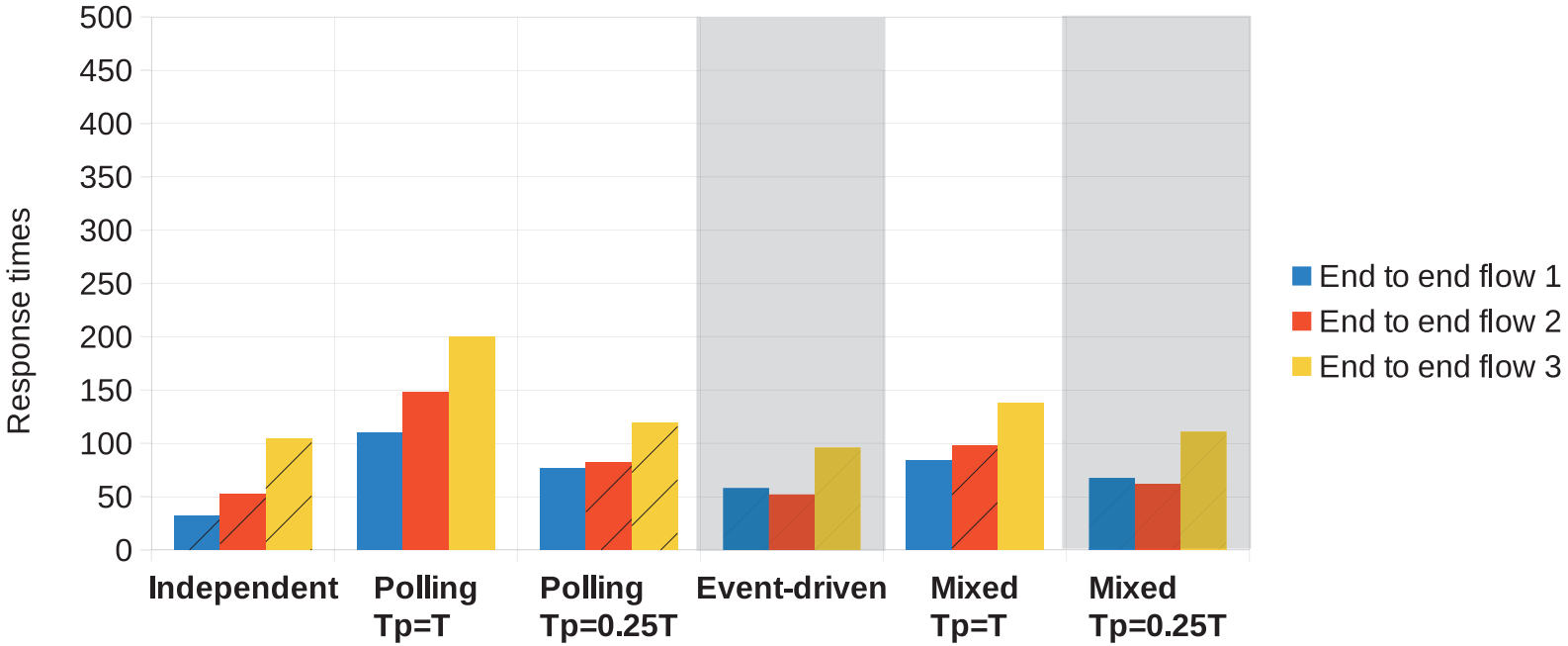
- Developed by Mäki-Turja and Nolin

Response time analysis (1/2)



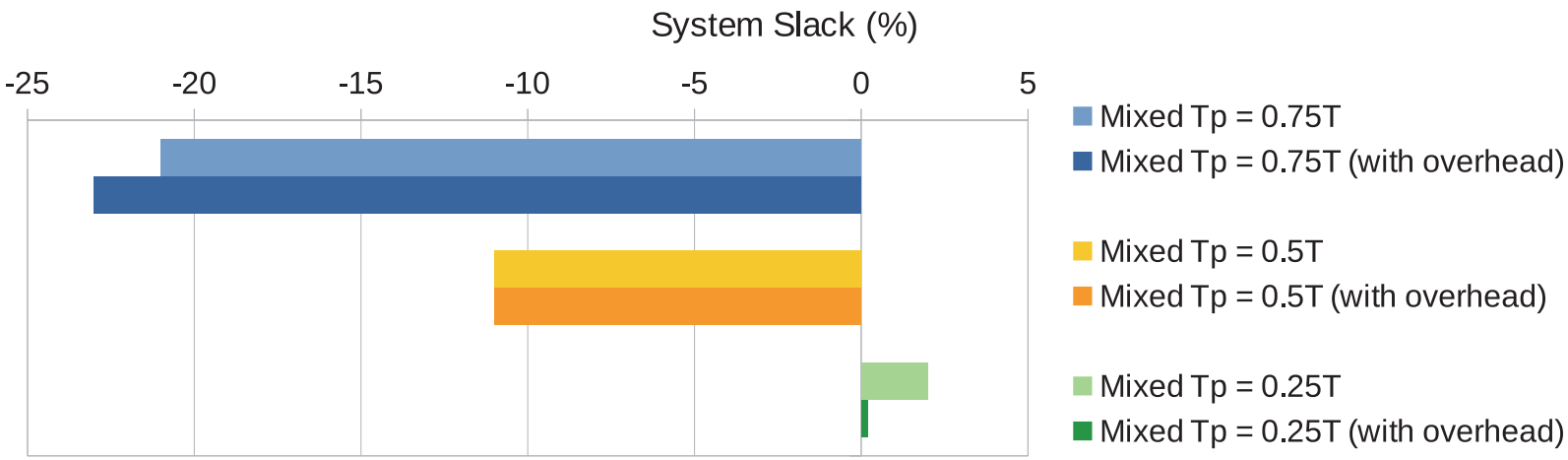
- **Results**
  - polling affects system schedulability
  - system schedulability improves when the polling period is reduced
- **No scenario with notion of event can be scheduled**

Response time analysis (2/2)



- Priority assignment optimized using the HOSPA algorithm
- Results
  - same conclusions but with better schedulability (except scenario 1)
  - one scenario based on polling is schedulable

Response time analysis (with overhead)



- **Polling overhead = 200μ**
- **Results**
  - **polling overhead may impact on system schedulability**
    - **tradeoff between polling period and polling overhead**

Performance analysis (1/2)

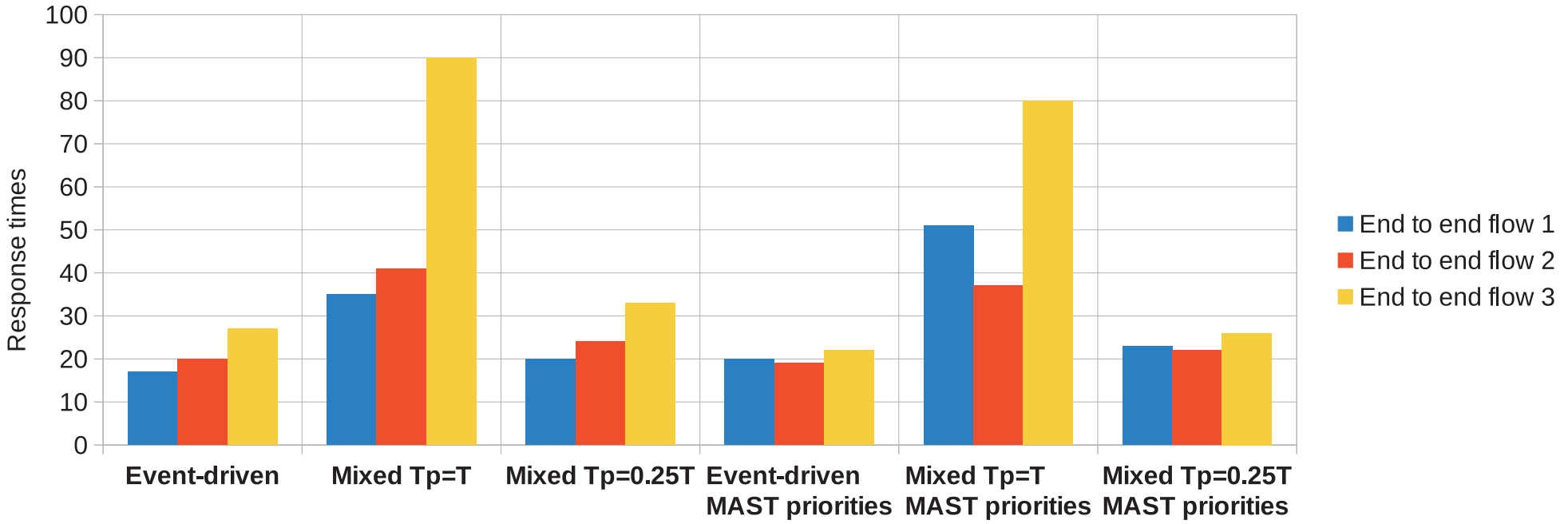
UNIVERSIDAD DE CANTABRIA

- Hardware platform
  - 3 embedded nodes
  - 800 Mhz
  - CAN bus
  - Philips SJA 1000 chip controller
- Ada software platform
  - MaRTE OS v 1.9
  - Synthetic task workloads
    - random values and execution-time clocks
  - Conditional entry call in a protected object to implement *polling*

Implementation issues:

- Tx buffer with capacity for a single message in CAN controller
  - priority inversion problem during bus arbitration process
  - Ada driver needs to replace the message in the tx buffer
- Lack of global clock to measure asynchronous e2e flows
  - external oscilloscope to measure the delay between digital signals

Performance analysis (2/2)



- average response times improves when the polling period is reduced
- event-driven model also obtains better average response times

Conclusions

UNIVERSIDAD DE CANTABRIA

Proposal of a real-time model for the analysis of polling tasks

- **no restriction in the choice of T_{poll}**
- **enables engineers to assess the effects of polling in their systems**

Using polling decreases system schedulability

- **higher response times (worst-case and average)**
- **polling overhead may have a non-negligible impact**

Priority assignment tool optimized all the scenarios

Integrated in the MAST toolsuite (open-source)

- **check it out at *http://mast.unican.es/***

everything will be okay

in the end.

if it's not okay,

it's not the end.

(unknown)