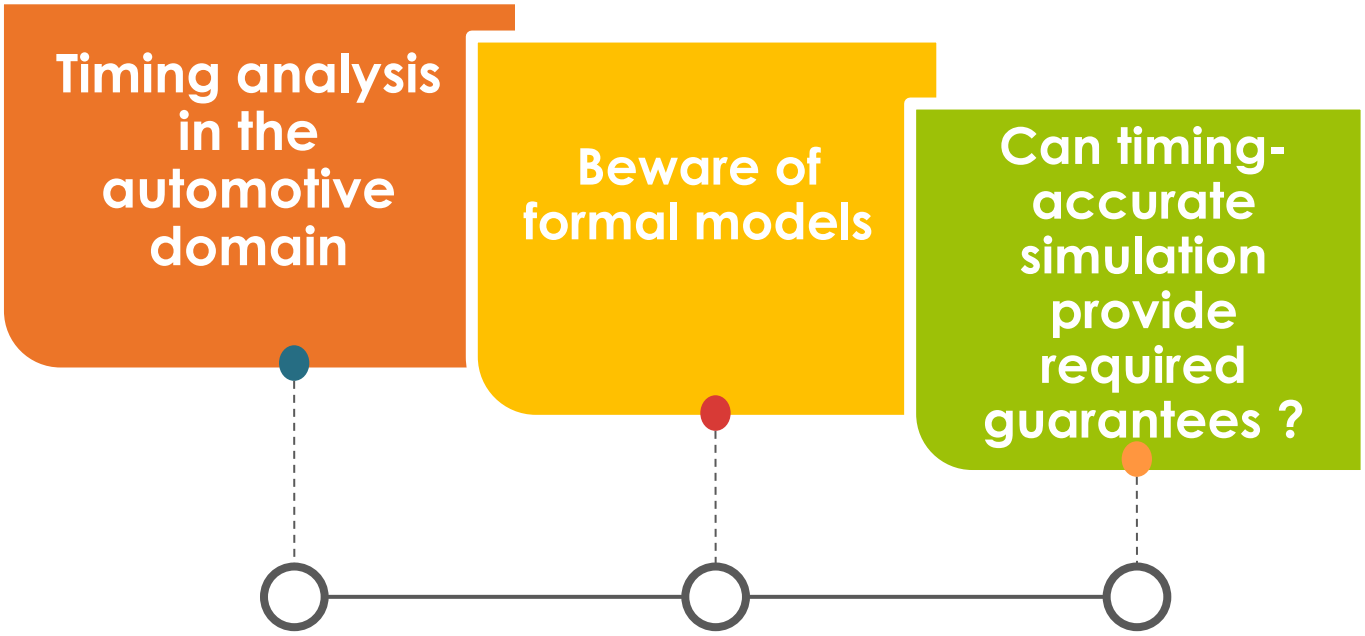
DESIGN, AUTOMATION & TEST IN EUROPE
14 – 18 March, 2016 · ICC · Dresden · Germany
The European Event for Electronic System Design & Test

Timing Analysis of Automotive Architectures and Software

Nicolas Navet
University of Luxembourg,
founder RealTime-at-Work

Outline

- ✓ **Focus is not the formalisms but on what to expect from timing analysis**



Software has become the key to innovation

- ✓ Software grows exponentially
- ✓ complex new technologies are introduced
- ✓ Pace of innovation ↑

How (formal) timing analysis can keep up?

Software is disrupting complete industries

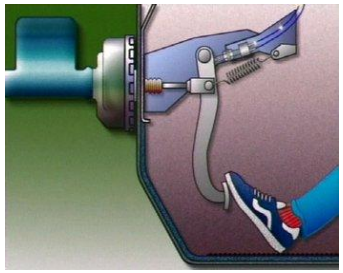
Every company has to learn to become a software company

Model-Driven Development is certainly a powerful enabler but ..

Still lacks

- ✓ Timing-augmented design flow
- ✓ Timing equivalent execution between model and run-time
- ✓ Automation features: “state the what, not the how” + “correct by construct”

Hundreds of timing constraints



- ✓Responsiveness
- ✓Freshness of data
- ✓Jitters
- ✓ Synchronicity
- ✓ …

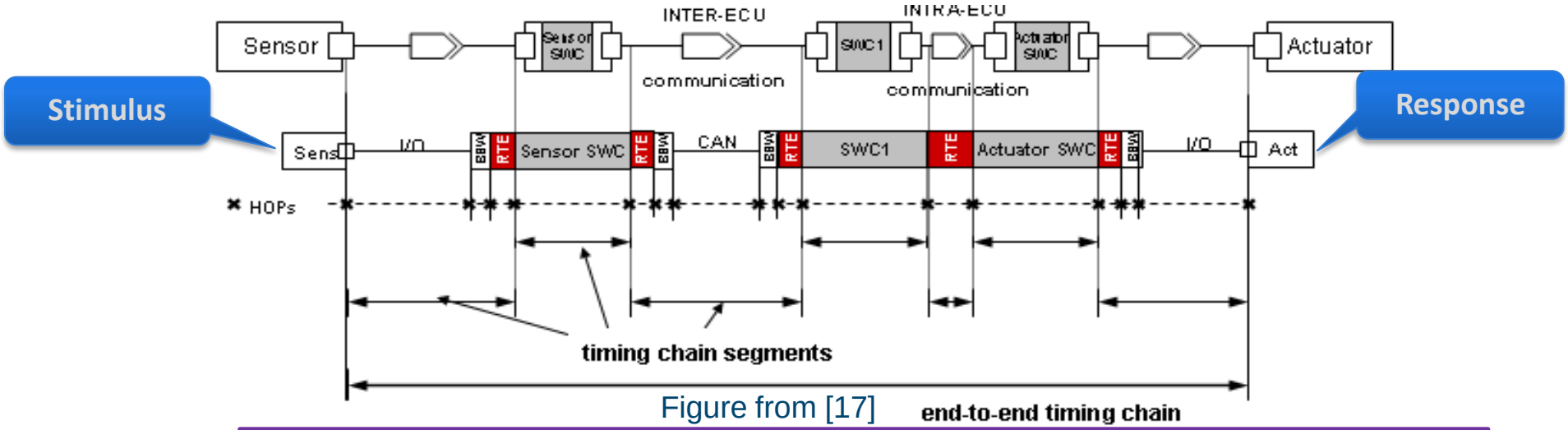


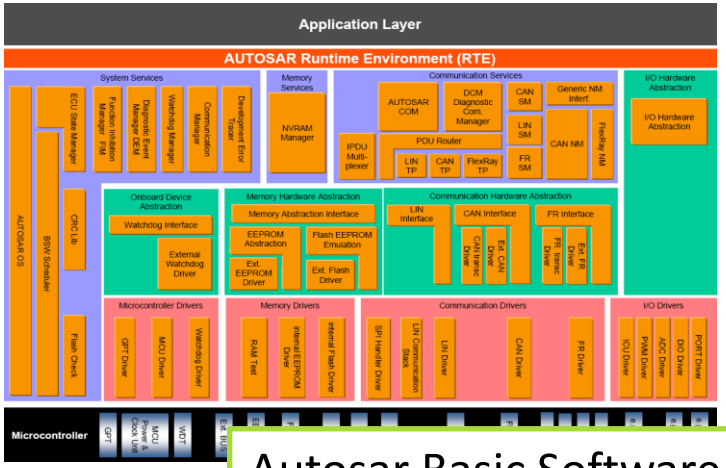
Figure from [17]

Involves hardware, software, networks, gateways, runtime environment (OS, Middleware, hypervisors)
Multi-source SW and HW

What makes things hard in automotive

Technologies: numerous, complex and not conceived with verifiability as a requirement

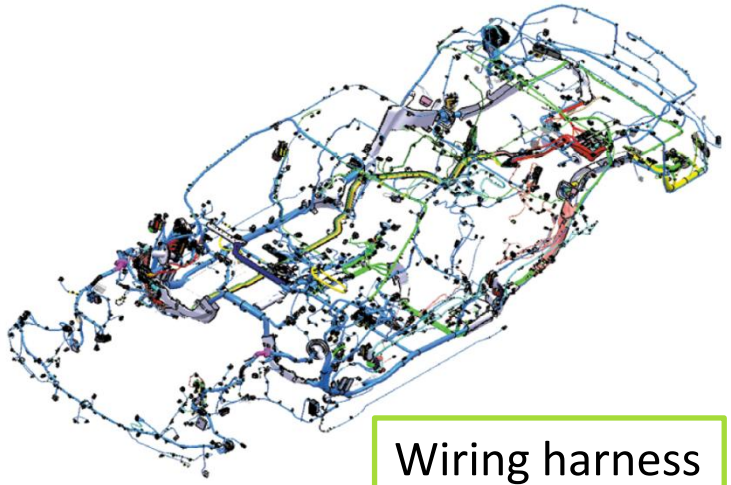
- ✓ **# of networks, complexity of Autosar (>150 doc) with limited support for timing specification, multi-core ECUs, GPU computing for ADAS, etc**
- ✓ **# of functional domains, buses, gateways, ECUs, size of code, tasks, wiring, number of variants, etc**



Autosar Basic Software

Development process

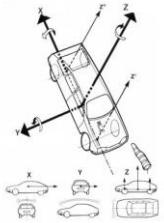
- ✓ **Limited regulatory constraints**
- ✓ **No "culture" of verification**
- ✓ **Traceability of timing constraints!**
- ✓ **Time, costs & resource utilization constraints**
- ✓ **Most developments are not done in-house**
- ✓ **Carry-over / Vehicle Family Management**



Wiring harness

Verification along the dev. cycle

$$K_i^k(t) \stackrel{\text{def}}{=} \underbrace{\left\lfloor \frac{J_i^k + \varphi_i^k(\phi^i)}{T_i^k} \right\rfloor}_{\text{max. number of instances that may accumulate at } t_c} + \underbrace{\left\lfloor \frac{t - \varphi_i^k(\phi^i)}{T_i^k} \right\rfloor + 1}_{\text{max. number of instances in } [t_c, t_c + t)} \quad (7)$$



Simulation

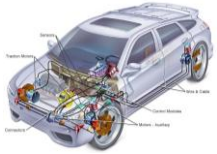
- ✓ **Functional simulation**
- ✓ **Timing -accurate simulation of ECU, bus, system-level**
- ✓ **Hardware in the loop, software-in-the-loop, processor in-the-loop, etc**

Formal verification

- ✓ **Worst-Case Execution Time analysis**
- ✓ **Worst-Case Response time analysis: ECU, bus, system-level**
- ✓ **Probabilistic analysis (academia)**

Testing

- ✓ **Integration tests**
- ✓ **Execution time measurements**
- ✓ **Off-line trace analysis**
- ✓ **Smart monitoring tools**



"Early stage" → Technological & design choices

"Project" → Configuration & optimization

"Real" → Refine and validate models & impact of non-conformance

Zoom on Worst-Case Response Times

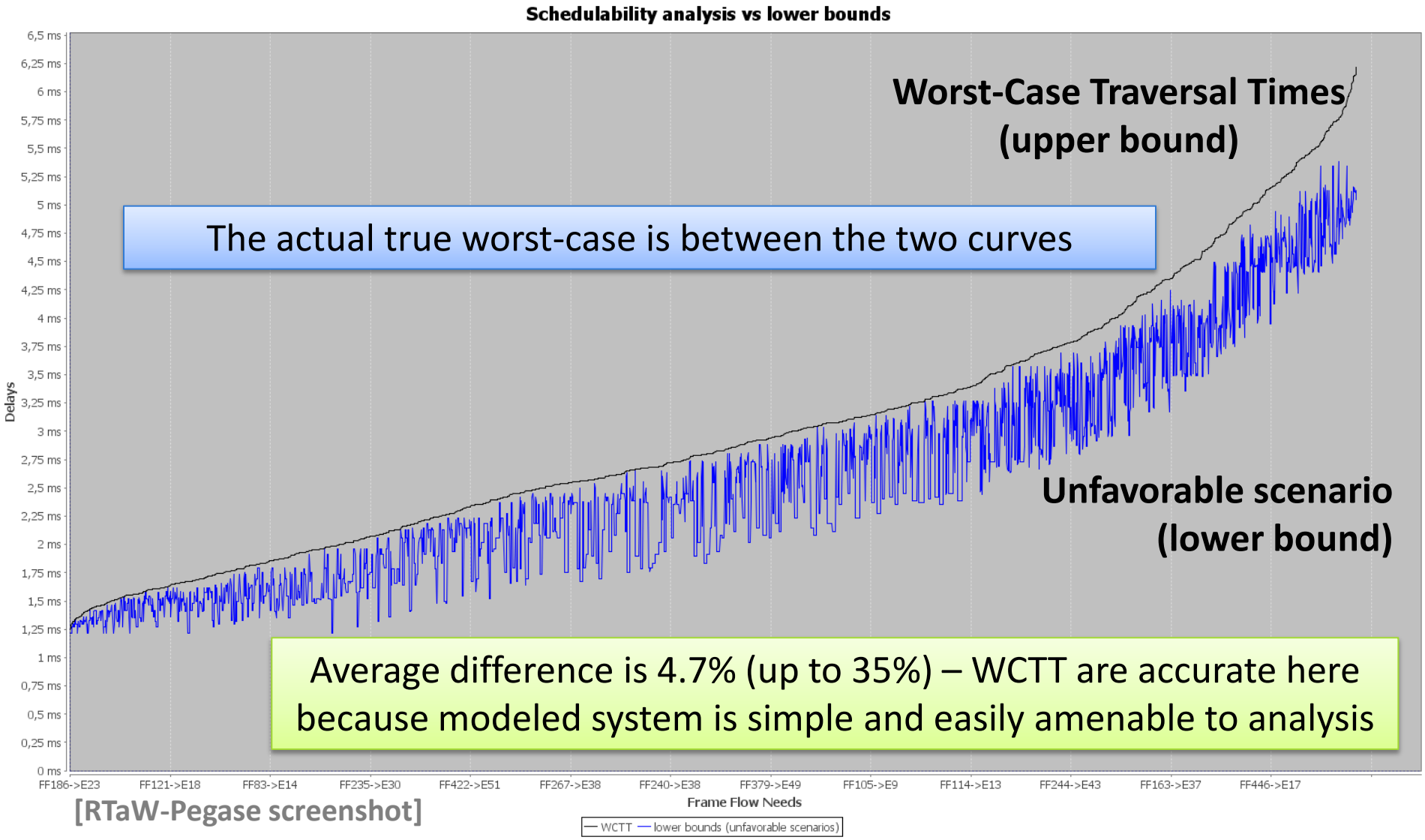
$$K_i^k(t) \stackrel{\text{def}}{=} \underbrace{\left\lfloor \frac{J_i^k + \varphi_i^k(\phi^i)}{T_i^k} \right\rfloor}_{\text{max. number of instances that may accumulate at } t_c} + \underbrace{\left\lfloor \frac{t - \varphi_i^k(\phi^i)}{T_i^k} \right\rfloor + 1}_{\text{max. number of instances in } [t_c, t_c + t)} \quad (7)$$

Simulation | **Formal verification** | **Testing**

WCRT: formalisms mature enough to derive usable bounds … if system complexity is "reasonable"

✓ **Worst-Case Response time analysis: ECU, networks, system-level**

Illustration: Network Calculus Ethernet analysis vs Lower-bounds [1]



[RTaW-Pegase screenshot]

Zoom on Worst-Case Response Times

$$K_i^k(t) \stackrel{\text{def}}{=} \left\lfloor \frac{J_i^k + \varphi_i^k(\phi^i)}{T_i^k} \right\rfloor + \left\lfloor \frac{t - \varphi_i^k(\phi^i)}{T_i^k} \right\rfloor + 1$$

Sim

Accurate model → verification

Approximate model →debugging, but usually unpredictably unsafe for verification

esting

- ✓ **Worst-Case Response time analysis: ECU, networks, system-level**

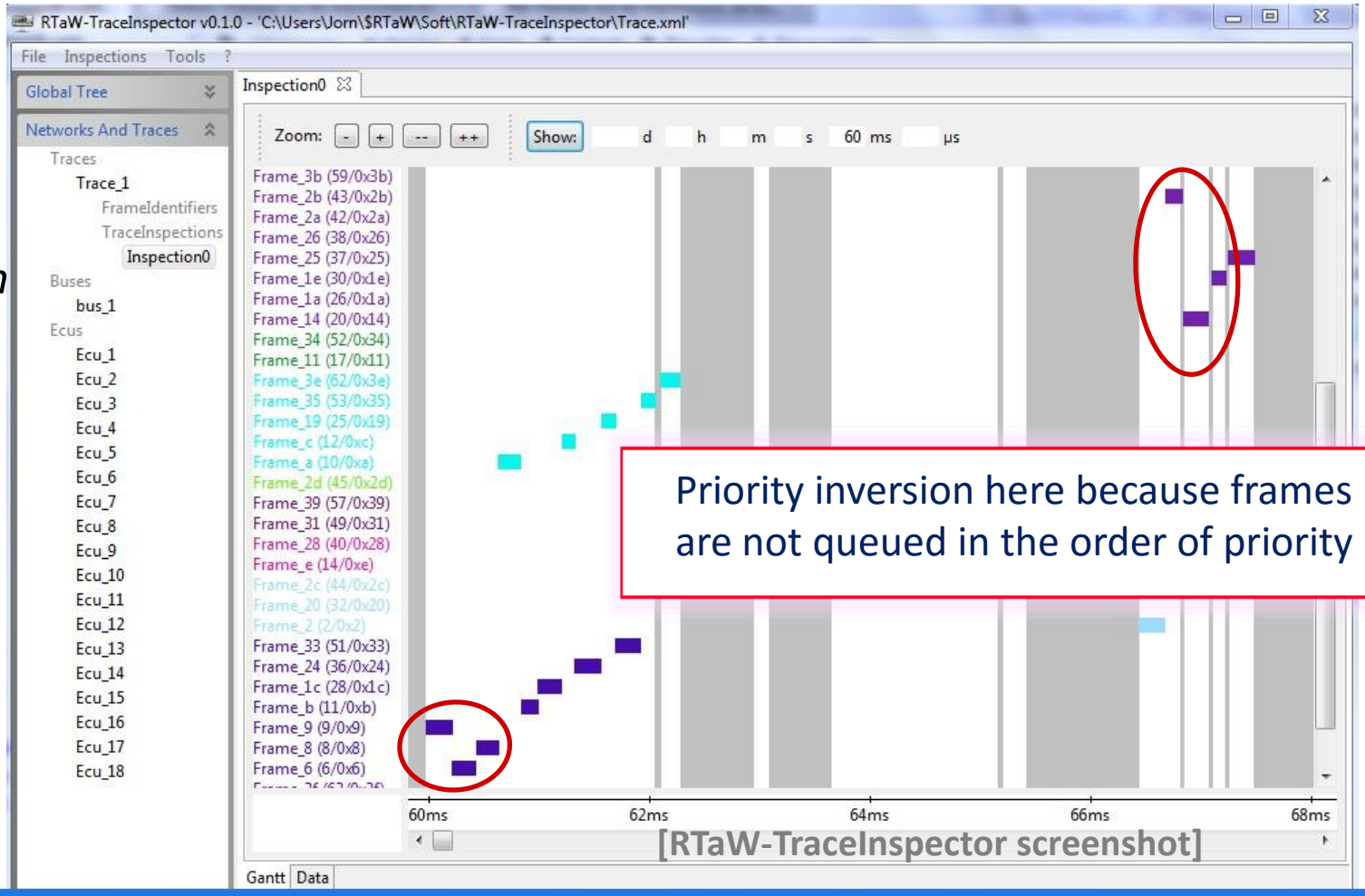
Requires knowledge of

- ✓ All activities: tasks, runnables, frames, signals
- ✓ Software code to derive execution times
- ✓ Complete embedded architecture with all scheduling & configuration parameters for buses and ECUs

Conservative assumptions possible with high resource utilization in automotive ?!

Models cannot replace testing

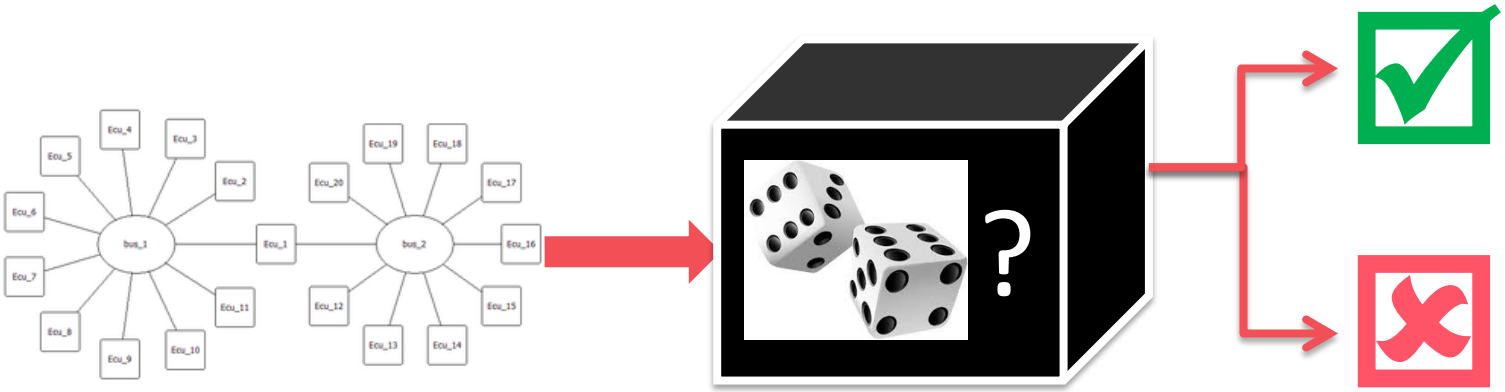
Ex: CAN communication traces



Priority inversion here because frames are not queued in the order of priority

Check comm. stack implementation, periods, offsets, jitters, model for aperiodic traffic and transmission errors, clock drifts, etc ..

Question: How do we know (formal) timing analysis models are trustworthy ?!



What do we have at hand

- ✓ Are the models published ? Usually no
- ✓ Is the source code of the tool available? No
- ✓ Do we have qualification ? No

Black-box tools

- ✓ Are there public benchmarks on which validate the results? No
- ✓ Limited number of end-users and cost-pressure ? Yes
- ✓ Complexity of the models and implementations? High
- ✓ Can we prove the correctness of the analysis results ? Not yet – step in that direction [2] for Network-Calculus analyses

Good practice - several techniques and several tools for cross-validation

Examples of cross-validations

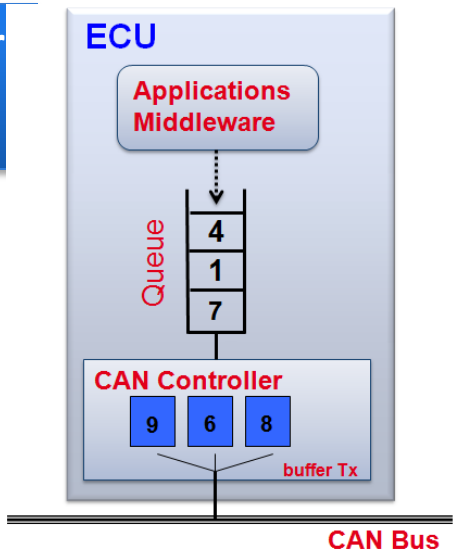
- ✓ **Comparing simulation and analysis results**
- ✓ **Validating a simulator using real communication/execution traces:** e.g., comparing inter-arrival times distributions
- ✓ **Re-simulating worst-case situation from schedulability analysis**
- ✓ **Validating schedulability analysis against lower-bounds:** e.g., validating Network-Calculus AFDX analysis with unfavorable scenarios from [3]
- ✓ **Cross-validating schedulability analysis by comparing different formalisms / tools:** e.g. network-calculus VS event-streams VS trajectory approach
- ✓ …

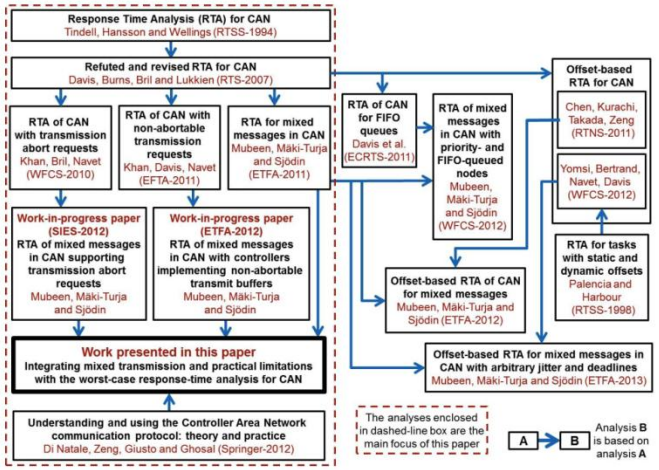
Validating timing accurate simulation models is much easier than schedulability analysis tools

Complex analytic models is a dead-end

Ex: Towards realistic Controller Area Network Analyses

- ✓ Non-prioritized waiting queues
- ✓ Non-abortable transmission requests
- ✓ Not enough transmission buffers
- ✓ Delays in refilling the buffers
- ✓ Delay data production / transmission request
- ✓ Segmented messages
- ✓ Autosar mixed-transmission requests
- ✓ Aperiodic traffic
- ✓ Transmission errors
- ✓ Gatewayed traffic
- ✓





Subset of the 50+ papers [14]

- ✓ Not everything covered, no complete integration
- ✓ Many analyses too pessimistic to be usable
- ✓ Precise analyses are often intractable and error prone

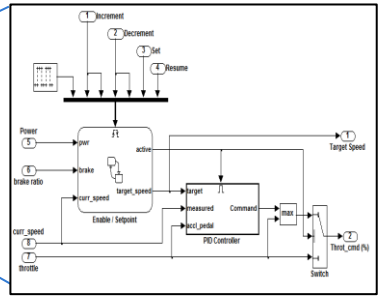
If formal analysis is needed, systems must be conceived accordingly

Timing-accurate simulation of embedded architectures

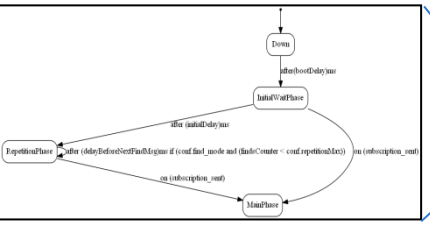
```
after(bootDelay)ms {

  offer_found = false;
  assert(conf.initialDelayMin >= 0);
  assert(conf.initialDelayMin <= conf.initialDelayMax);
  /* find_msg = {conf.id, conf.expected_service, FIND}; */
  find_msg.source_instance_id = conf.instance_id;
  find_msg.destination_instance_id = 0; /* multicast */
  find_msg.service_id = conf.expected_service;
  find_msg.kind = FIND;

  /* Pick a random initial delay */
  initialDelay = rand_uniform(conf.initialDelayMin, conf.i
  println("client initial delay: %u\n", initialDelay);
} to InitialWaitPhase;
```

Application software

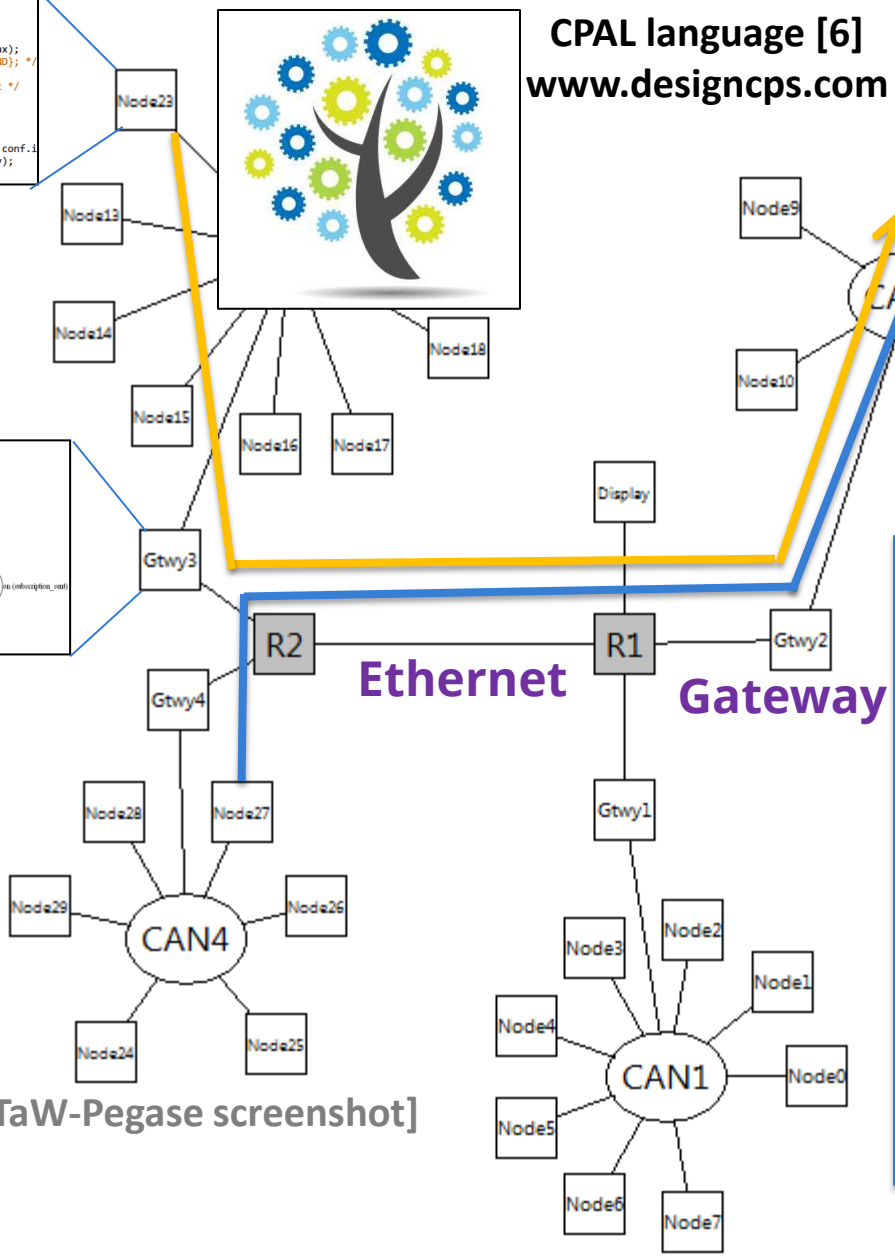
CPAL language [6]
www.designcps.com



Functional model



High-level protocol layer

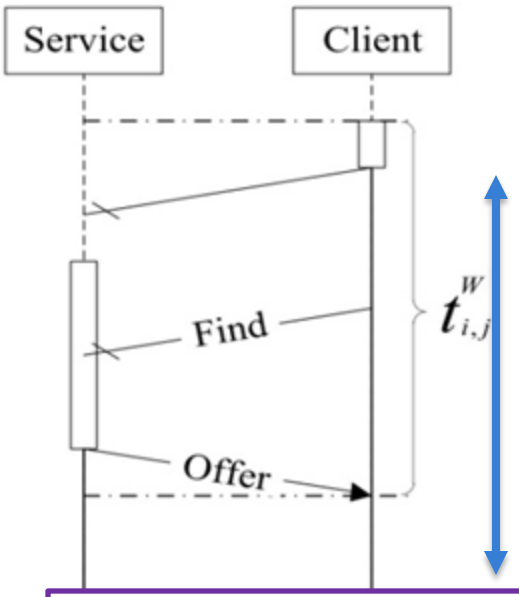


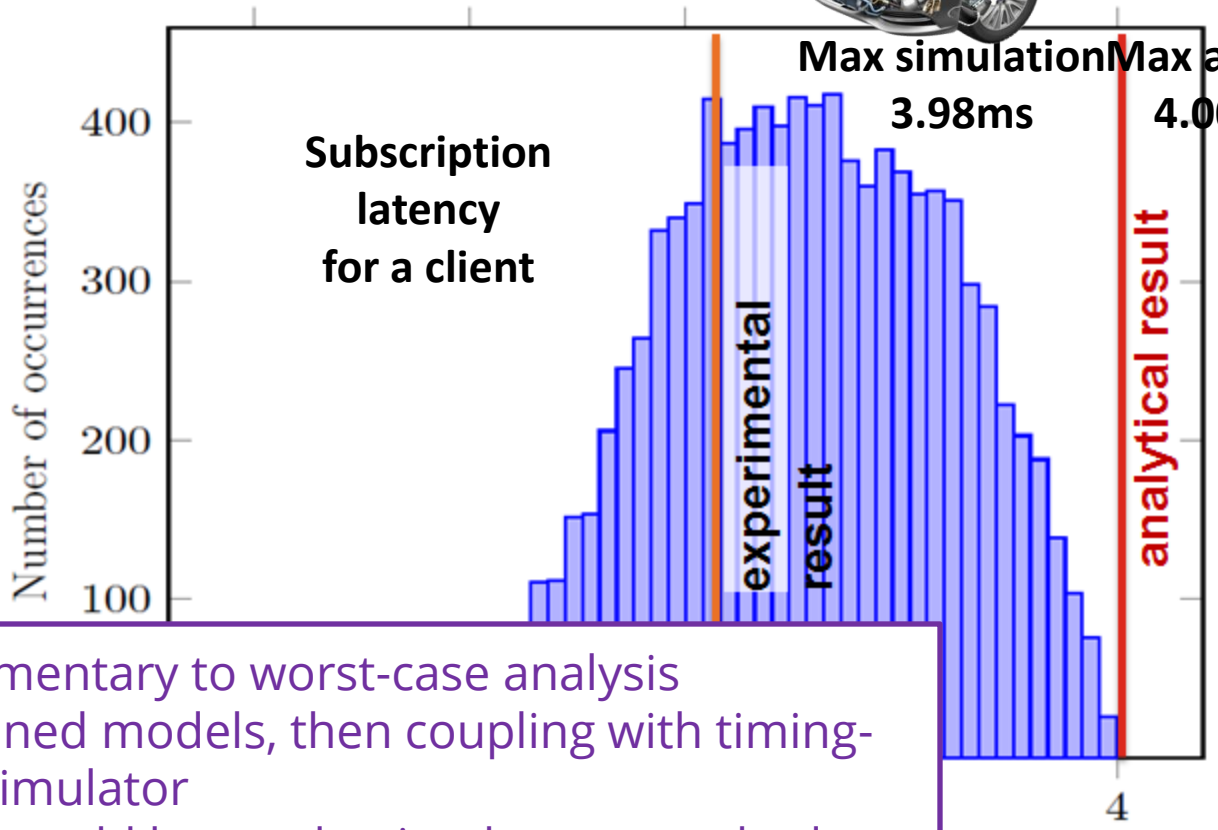
[RTaW-Pegase screenshot]

- ✓Today: timing accurate simulation /analysis of complete heterogeneous embedded architectures
- ✓ Speedup > 10
- ✓Suited up to (1-10-^{6}) quantiles
- ✓Tomorrow: system-level simulation with models of the *functional* behavior

Timing analysis: Some/IP SD [7,8]

SOME/IP SD: **service discovery for automotive Ethernet**
Objective: find the right tradeoff between subscription latency and SOME/IP SD overhead





- ✓ Simulation complementary to worst-case analysis
- ✓ 2 steps: coarse grained models, then coupling with timing-accurate network simulator
- ✓ Same CPAL models could be used to implement testbeds

Simulation for .. safety-critical systems ?!

IMO: if system can be made robust to rare (quantified) deadline misses, then designing with simulation is more effective in terms of resource usage

Know what to expect from simulation – typically:

- ✓ Worst-case behaviors are out of reach but extremely rare events (e.g., Pr << 10^{-6} - see[1])
- ✓ Able to provide guarantees for events up Pr < 10^{-6} in a few hours
- ✓ Coarse-grained lower-bounds analysis to cross-validate

Sound simulation methodology

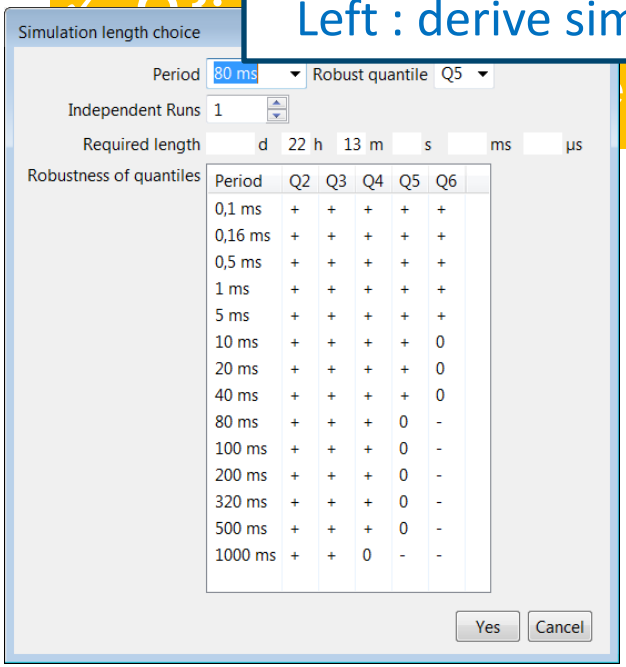
- ✓ **Q1:** is a single run enough ?
- ✓ **Q2:** can we run simulation in parallel and aggregate results ?
- ✓ **Q3:** simulation length ?
- ✓ **Q4:** correlations between "feared events" ?

Simulation for .. safety-critical systems ?!

Simulation methodology

- ✓ Q1: is a single run s
- ✓ Q2:
- ✓ Q3:

Tool support should help here:
Right : numbers in gray should not be trusted
Left : derive simulation time wrt target quantile



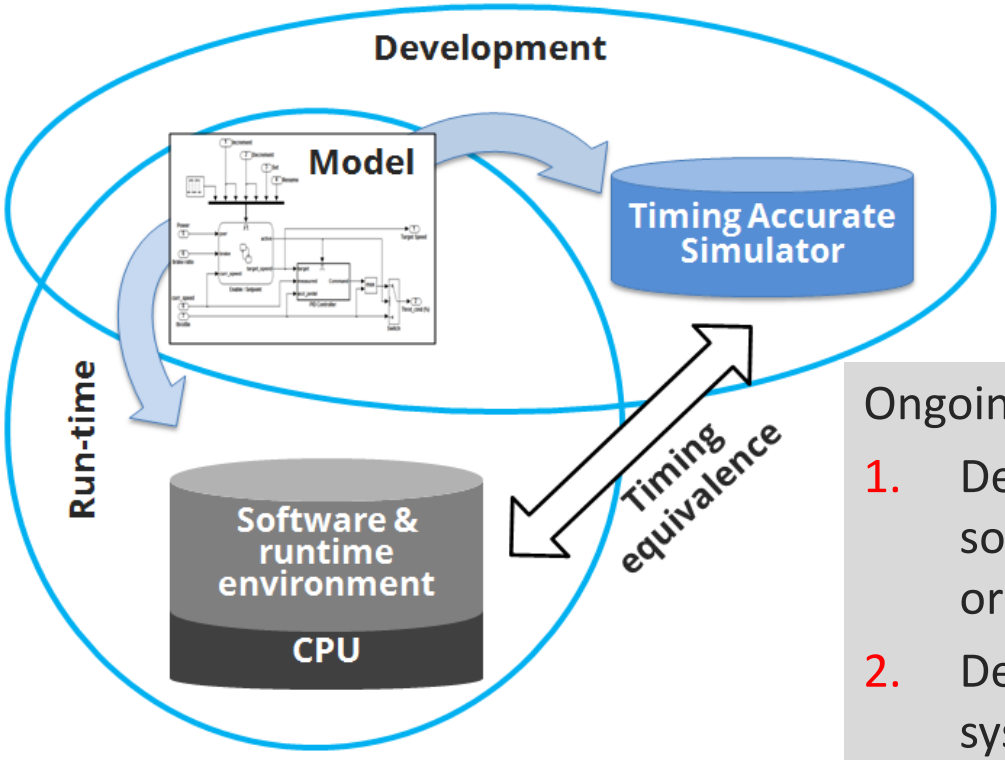
Period	Q2	Q3	Q4	Q5	Q6
0,1 ms	+	+	+	+	+
0,16 ms	+	+	+	+	+
0,5 ms	+	+	+	+	+
1 ms	+	+	+	+	+
5 ms	+	+	+	+	+
10 ms	+	+	+	+	0
20 ms	+	+	+	+	0
40 ms	+	+	+	+	0
80 ms	+	+	+	0	-
100 ms	+	+	+	0	-
200 ms	+	+	+	0	-
320 ms	+	+	+	0	-
500 ms	+	+	+	0	-
1000 ms	+	+	0	-	-

Min	Average	Q2	Q3	Q4	Q5	Q6	Max	Bound
						477 ms	0,477 ms	0,550 ms
						719 ms	0,719 ms	0,830 ms
						925 ms	0,925 ms	1,074 ms
						167 ms	1,167 ms	1,354 ms
						943 ms	0,943 ms	1,092 ms
						185 ms	1,185 ms	1,372 ms
						427 ms	1,427 ms	1,652 ms
						669 ms	1,669 ms	1,932 ms
						1,339 ms	1,339 ms	1,564 ms
0,148 ms	0,242 ms	0,979 ms	1,382 ms	1,643 ms	1,791 ms	1,811 ms	1,822 ms	2,124 ms
0,218 ms	0,313 ms	1,061 ms	1,481 ms	1,750 ms	1,875 ms	2,009 ms	2,036 ms	2,386 ms
0,522 ms	0,686 ms	1,490 ms	1,897 ms	2,116 ms	2,267 ms	2,388 ms	2,509 ms	4,890 ms
0,450 ms	0,615 ms	1,398 ms	1,811 ms	2,104 ms	2,293 ms	2,402 ms	2,672 ms	4,818 ms
0,720 ms	0,929 ms	1,832 ms	2,128 ms	2,280 ms	2,374 ms	2,486 ms	2,515 ms	2,946 ms
0,702 ms	0,887 ms	1,897 ms	2,280 ms	2,544 ms	2,573 ms	2,710 ms	2,756 ms	3,470 ms
0,236 ms	0,367 ms	1,423 ms	2,032 ms	2,347 ms	2,618 ms	2,710 ms	2,863 ms	3,750 ms
0,962 ms	1,271 ms	2,374 ms	2,664 ms	2,904 ms	2,989 ms	3,166 ms	3,254 ms	4,030 ms
0,720 ms	0,957 ms	1,986 ms	2,374 ms	2,588 ms	2,773 ms	2,854 ms	2,941 ms	3,750 ms
0,112 ms	0,281 ms	1,643 ms	2,280 ms	2,618 ms	2,854 ms	2,989 ms	3,103 ms	4,186 ms
0,166 ms	0,252 ms	1,043 ms	1,481 ms	1,801 ms	2,092 ms	2,153 ms	2,238 ms	3,276 ms
0,166 ms	0,338 ms	1,710 ms	2,307 ms	2,633 ms	2,854 ms	2,971 ms	3,060 ms	4,396 ms
1,168 ms	1,567 ms	2,695 ms	2,989 ms	3,202 ms	3,277 ms	3,373 ms	3,460 ms	4,640 ms
0,236 ms	0,421 ms	1,963 ms	2,603 ms	2,921 ms	3,076 ms	3,221 ms	3,239 ms	4,640 ms
0,522 ms	0,801 ms	2,402 ms	3,023 ms	3,471 ms	3,698 ms	3,806 ms	3,871 ms	8,946 ms
0,702 ms	0,987 ms	2,515 ms	2,989 ms	3,258 ms	3,412 ms	3,483 ms	3,483 ms	4,920 ms
0,702 ms	0,987 ms	2,515 ms	2,989 ms	3,315 ms	3,491 ms	3,864 ms	3,864 ms	4,920 ms
0,302 ms	0,524 ms	2,092 ms	2,633 ms	2,954 ms	3,129 ms	3,181 ms	3,181 ms	4,744 ms
0,702 ms	0,989 ms	2,515 ms	2,989 ms	3,239 ms	3,451 ms	3,548 ms	3,548 ms	4,920 ms
0,218 ms	0,427 ms	2,080 ms	2,773 ms	3,166 ms	3,392 ms	3,532 ms	3,532 ms	5,182 ms
0,148 ms	0,345 ms	1,941 ms	2,648 ms	3,129 ms	3,315 ms	3,336 ms	3,336 ms	5,094 ms
0,182 ms	0,390 ms	2,056 ms	2,741 ms	3,166 ms	3,431 ms	3,817 ms	3,817 ms	6,718 ms
0,236 ms	0,444 ms	2,116 ms	2,773 ms	3,184 ms	3,511 ms	3,733 ms	3,733 ms	6,772 ms
0,218 ms	0,426 ms	2,092 ms	2,773 ms	3,202 ms	3,471 ms	3,587 ms	3,587 ms	6,754 ms
0,182 ms	0,391 ms	2,068 ms	2,726 ms	3,148 ms	3,412 ms	3,578 ms	3,578 ms	6,718 ms
0,166 ms	0,383 ms	2,080 ms	2,805 ms	3,184 ms	3,416 ms		3,416 ms	6,982 ms

[RTaW-pegase screenshot]

Timing-Augmented Model Driven Development

- ✓ **Functional integration fails if control engineering assumptions not met at run-time: sampling jitters, varying response times, etc**



Solution: injecting delays in the simulation - but how to do that early stage without knowledge of complete configuration ?

Ongoing work [6,18]:

1. Designer defines timing-acceptable solution in terms of significant events: order & quantified relationships btw them
2. Derive QoS needed from the runtime systems: CPU, comm. latencies
3. Resource reservation & QoS ensured at run-time

Key takeaways (1/2)

- ✓ **Body of efficient formalisms & tools but**
  - **models and their assumptions should be questioned by end-users**
  - **cross-validation is a must**
- ✓ **Ahead of us:**
  - **lower-bounds with search intensive techniques**
  - **better practices: validation benchmarks & proofs of result correctness**
  - **Mixed-criticality (MC) timing analyses for MC constraints**
- ✓ **Formal timing models cannot be safely used in systems that have not been conceived for timing analyzability → input for upcoming standards**

Key takeaways (2/2)

- ✓ **Timing-accurate simulation is well suited to automotive systems that can tolerate deadline misses with a *controlled* risk**
- ✓ **Today: timing accurate simulation of complete heterogeneous automotive communication architectures**
- ✓ **Tomorrow: system-level simulation with models of the functional behavior**
- ✓ **Formal methods most useful if 1) automated 2) integrated with standard development environments**
  → need for timing-augmented MDD with correct by construct system synthesis

References

[1] N. Navet, J. Seyler, J. Migge, "Timing verification of real-time automotive Ethernet networks: what can we expect from simulation?", Embedded Real-Time Software and Systems (ERTS 2016), Toulouse, France, January 27-29, 2016.

[2] E. Mabille, M. Boyer, L. Fejoz, and S. Merz, "Certifying Network Calculus in a Proof Assistant", 5th European Conference for Aeronautics and Space Sciences (EUCASS), Munich, Germany, 2013.

[3] H. Bauer, J.-L. Scharbarg, C. Fraboul, "Improving the Worst-Case Delay Analysis of an AFDX Network Using an Optimized Trajectory Approach", IEEE Transactions on Industrial informatics, Vol 6, No. 4, November 2010.

[4] CPAL – the Cyber-Physical Action Language, freely available from http://www.designcps.com, 2015.

[5] N. Navet, S. Louvart, J. Villanueva, S. Campoy-Martinez, J. Migge, "Timing verification of automotive communication architectures using quantile estimation", Embedded Real-Time Software and Systems (ERTS 2014), Toulouse, France, February 5-7, 2014.

[6] N. Navet N., L. Fejoz L., L. Havet , S. Altmeyer, "Lean Model-Driven Development through Model-Interpretation: the CPAL design flow", Technical report from the University of Luxembourg, to be presented at ERTSS2016, October 2015.

[7] J. Seyler, N. Navet, L. Fejoz, "Insights on the Configuration and Performances of SOME/IP Service Discovery", in SAE International Journal of Passenger Cars- Electronic and Electrical Systems, 8(1), 124-129, 2015.

[8] J. Seyler, T. Streichert, M. Glaß, N. Navet, J. Teich, "Formal Analysis of the Startup Delay of SOME/IP Service Discovery", Design, Automation and Test in Europe (DATE2015), Grenoble, France, March 13-15, 2015.

References Continued

[13] J. Rushby, Tutorial Introduction to Modern Formal Methods, available online.

[14] S. Mubeen, J. Mäki-Turja, M. Sjödin, “Integrating mixed transmission and practical limitations with the worst-case response-time analysis for Controller Area Network”, Journal of Systems and Software, Volume 99, 2015.

[15] N. Navet, F. Simonot-Lion, editors, “The Automotive Embedded Systems Handbook”, Industrial Information Technology series, CRC Press / Taylor and Francis, ISBN 978-0849380266, December 2008.

[16] P. Wallin, Axelsson, A Case Study of Issues Related to Automotive E/E System Architecture Development, IEEE International Conference and Workshop on the Engineering of Computer Based Systems, 2008.

[17] AUTOSAR, “Specification of Timing Extensions”, Release 4.0 Rev 2, 2010.

[18] S. Altmeyer, N. Navet, "Towards a declarative modeling and execution framework for real-time systems", First IEEE Workshop on Declarative Programming for Real-Time and Cyber-Physical Systems, San-Antonio, USA, December 1, 2015.