The use of simulation in the design of critical embedded systems

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

RTaW
RealTime-at-Work

uni.lu

NAFEMS Conference
Paris, June 9th 2016

Critical systems are often *very* complex

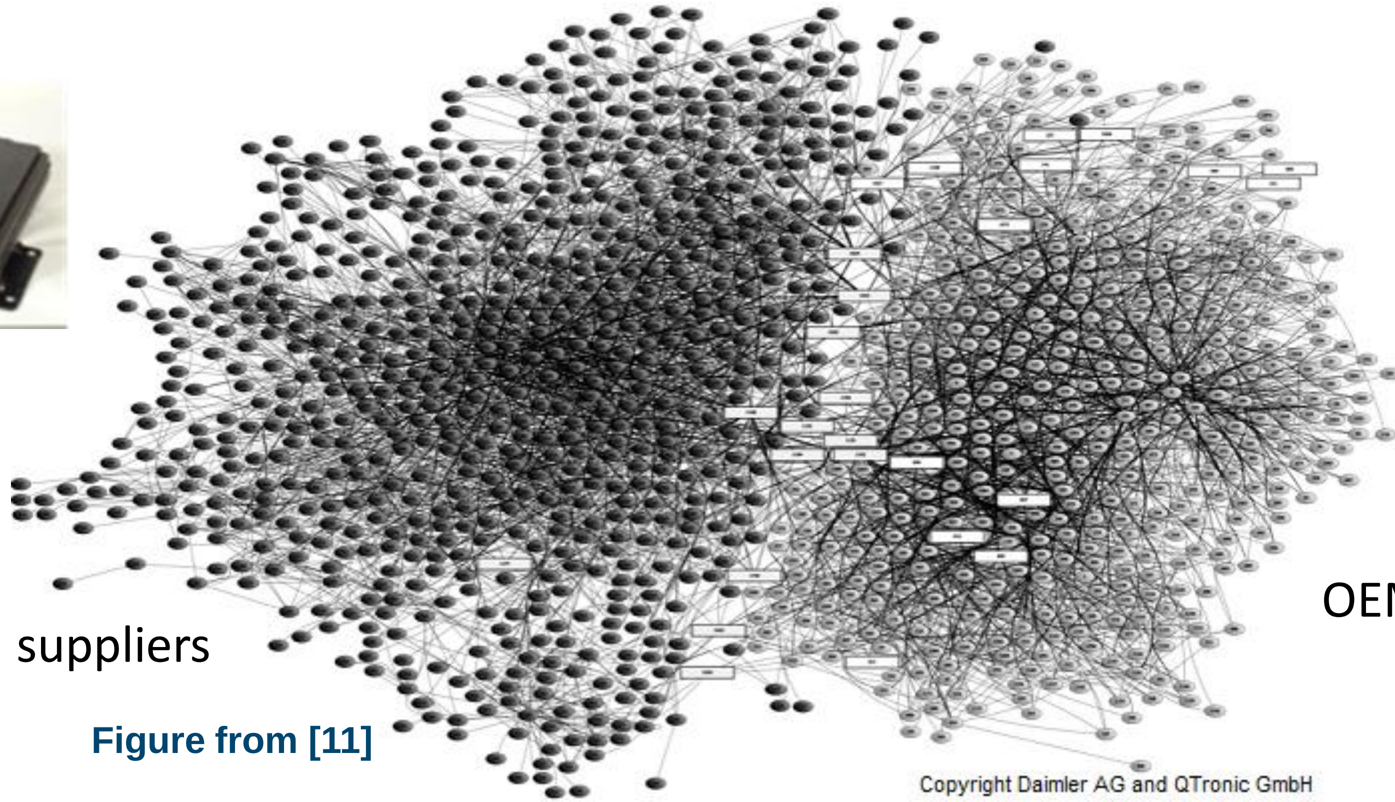
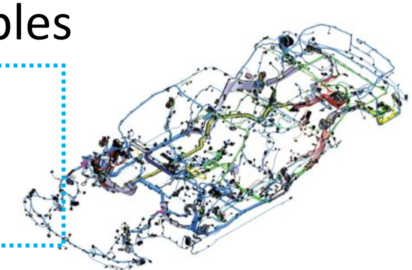


Figure from [11]

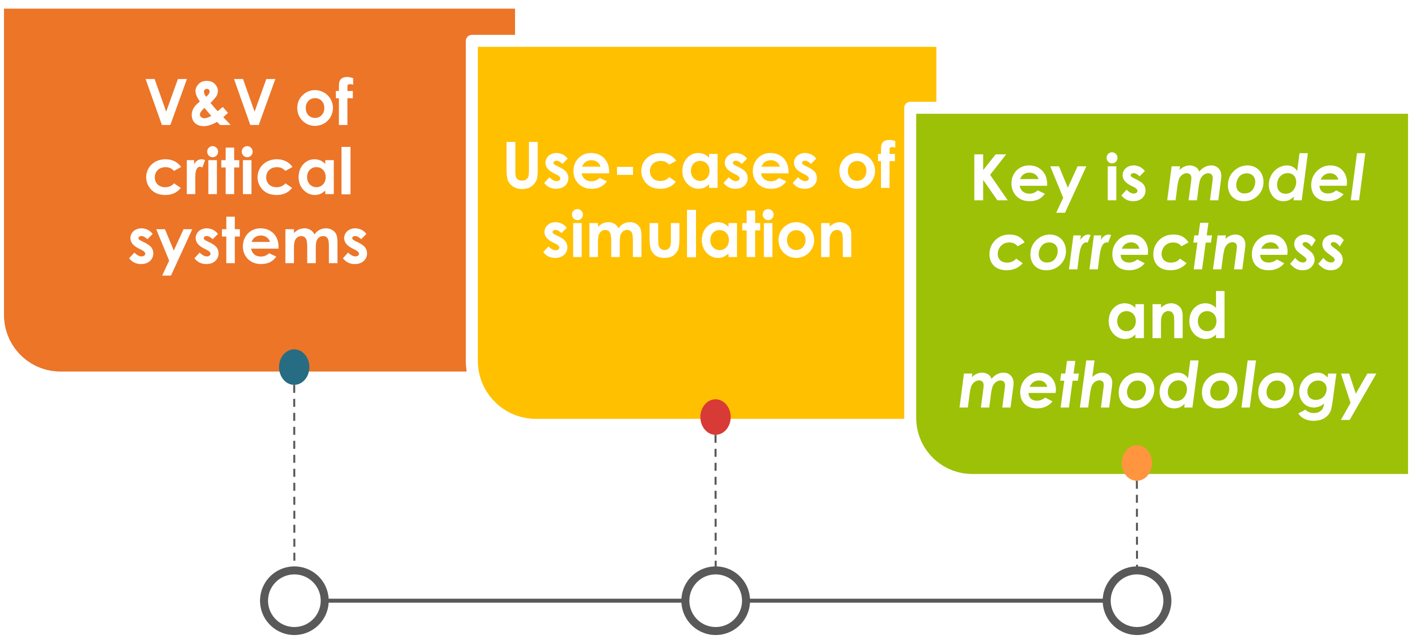
Inside an engine ECU: functions are the nodes (≈1500), edges are function calls, Functions are processing around 35000 variables

Complete Electrical and Electronic architecture: 10s of ECUs, many wired and some wireless networks, gateways, etc



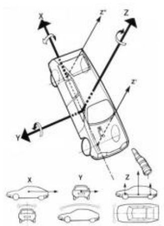
Outline

✓ **Simulation in the design of critical systems with a focus on timing-accurate simulation**



Verification along the dev. cycle

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



Simulation

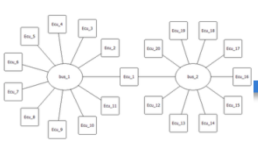
- ✓ Functional simulation
- ✓ Software-in-the-loop, hardware in the loop, etc
- ✓ Timing-accurate simulation of ECU, bus, system-level

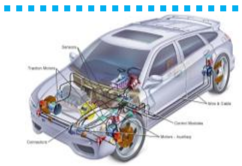
Formal verification

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

Testing

- ✓ Execution time measurements
- ✓ Integration tests
- ✓ Off-line trace analysis & monitoring tools
- ✓ …





"Early stage" — Technological & design choices

"Project" — Configuration & optimization

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

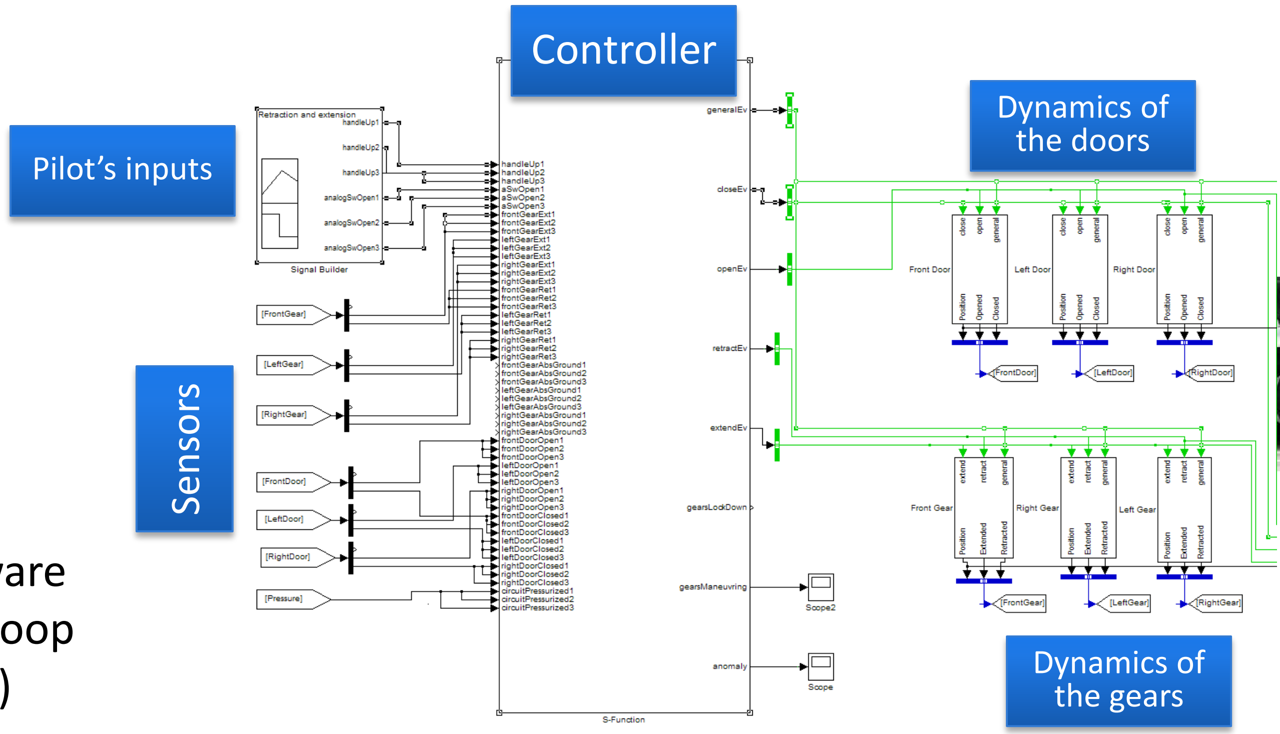
Critical systems are often *real-time systems*

- ✓ **Correctness in the *value domain* → functional simulation**

Model in-the-loop (Mil)

Software in-the-loop (Sil)

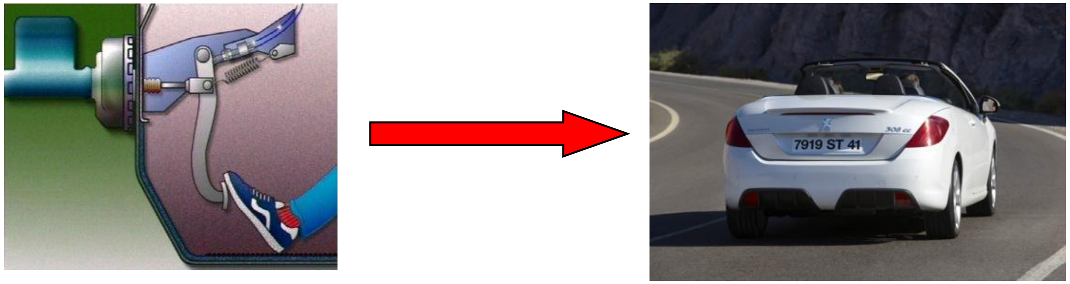
Hardware in-the-loop (Hil)



airplane landing gear [9]

- ✓ **Correctness in the *time domain* → timing accurate simulation, everything else is abstracted away**

Hundreds of timing constraints



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

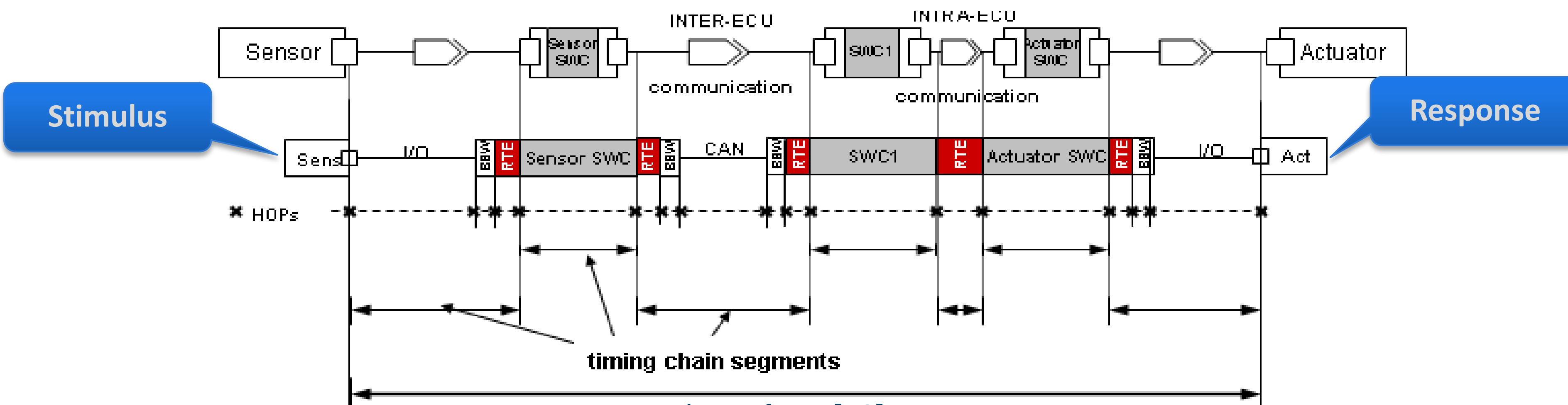


Figure from [10]

Timing-accurate simulation: the activities of the system are modelled by their activation patterns and execution time – functional behaviour is not captured

Zoom on response time constraints

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

Sim… | …esting

Accurate model → verification

Approximate model → debugging, but usually unpredictably unsafe for verification

- ✓ **Response times by simulation: ECU, networks, system-level**

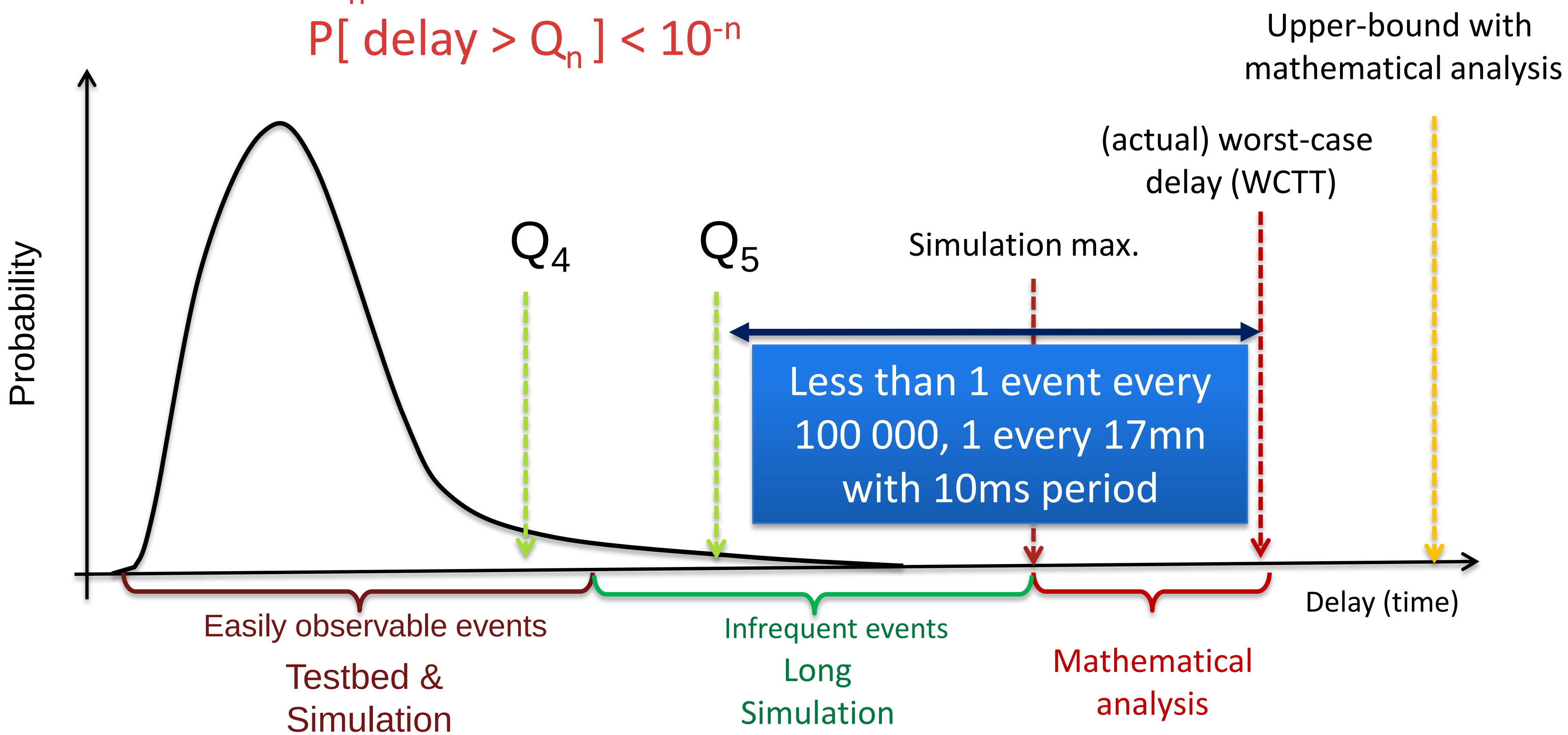
Requires knowledge of

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

Solution for early-stage verification: conservative assumptions and time budget per resource

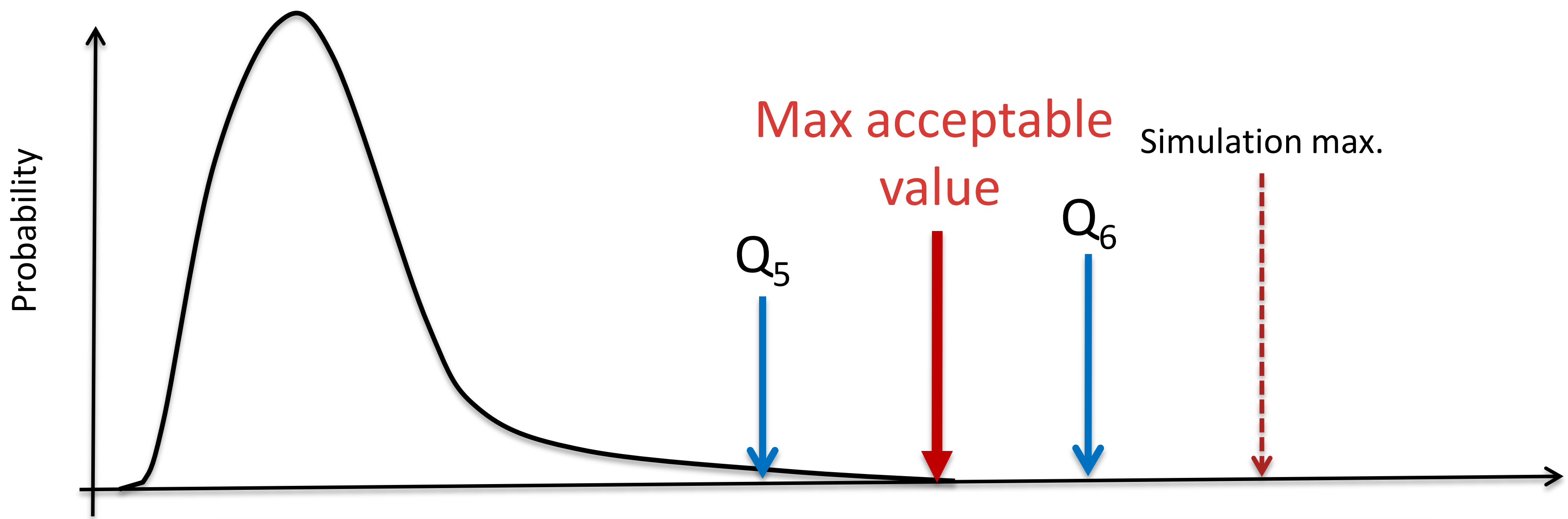
Interest in the tails of the distribution

Quantile Q_n: smallest value such that $P[\text{ delay} > Q_n] < 10^{-n}$



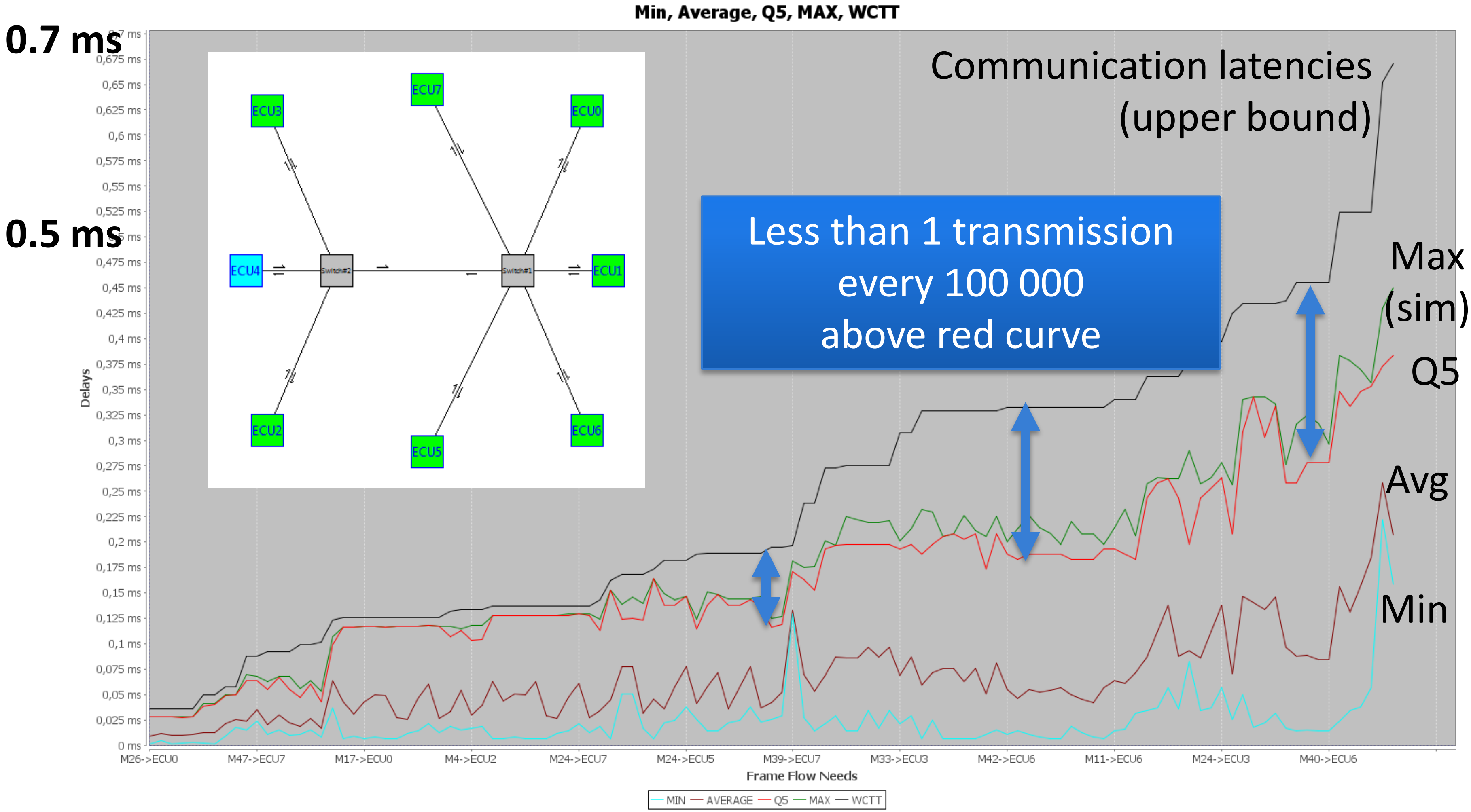
Using simulation means accepting a quantified risk - system must be robust to that

Working with quantiles in practice – see [5]



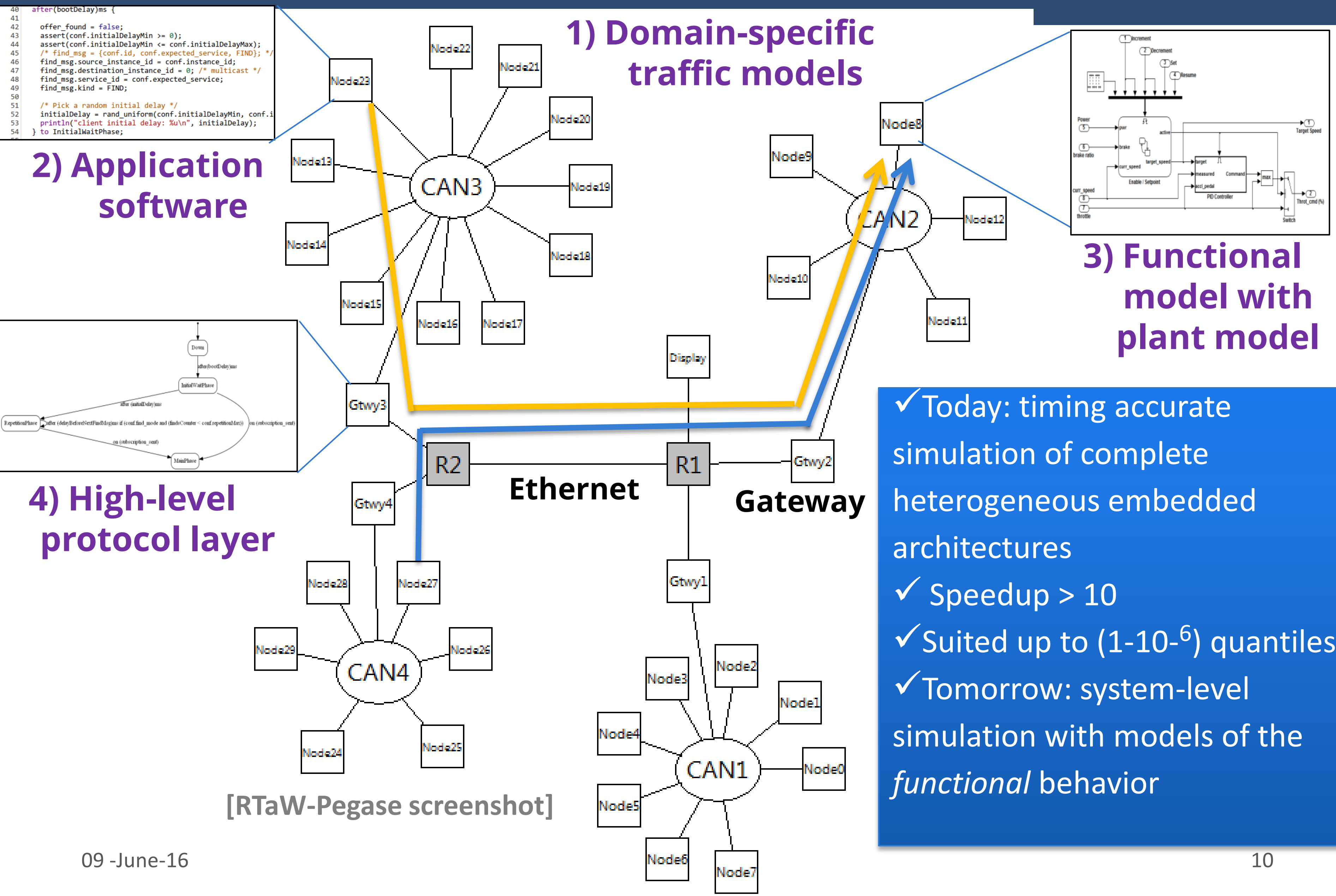
1. Identify frame deadline
2. Decide the tolerable risk → target quantile
3. Simulate "sufficiently" long
4. If target quantile value is below max. acceptable value, performance objective is met

Performance metrics: illustration on a Daimler prototype network (ADAS, control functions) [1]



The 58 flows of data sorted by increasing communication latencies

Simulation of embedded architectures



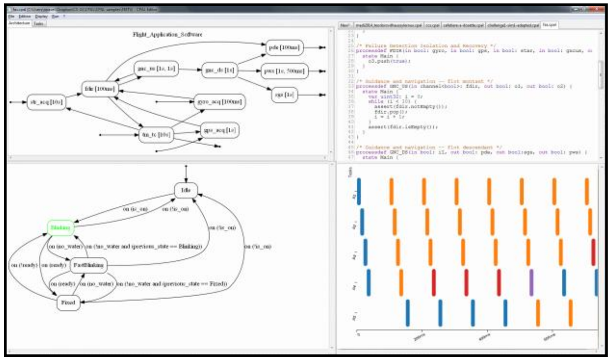
[RTaW-Pegase screenshot]

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

CPAL simulation language – see [4]

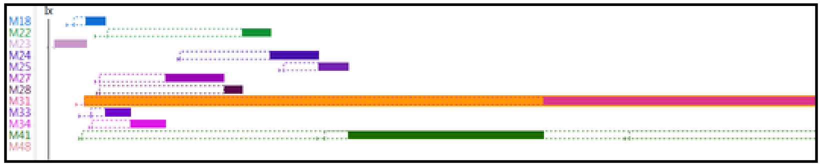
1 **Model and program**

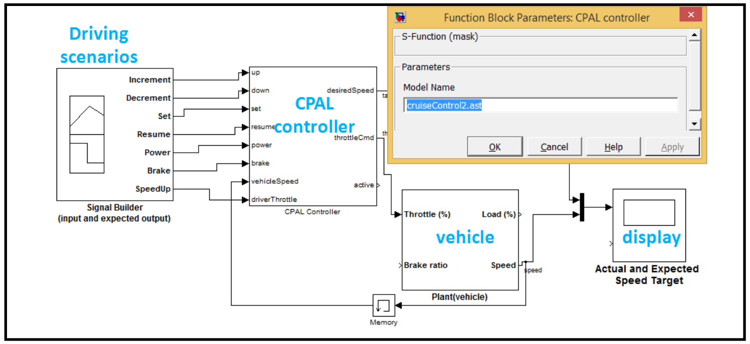
functional and non-functional concerns



2 **Simulate**

possibly embedded within external tools such as RTaW-Pegase™ and Matlab/Simulink ™



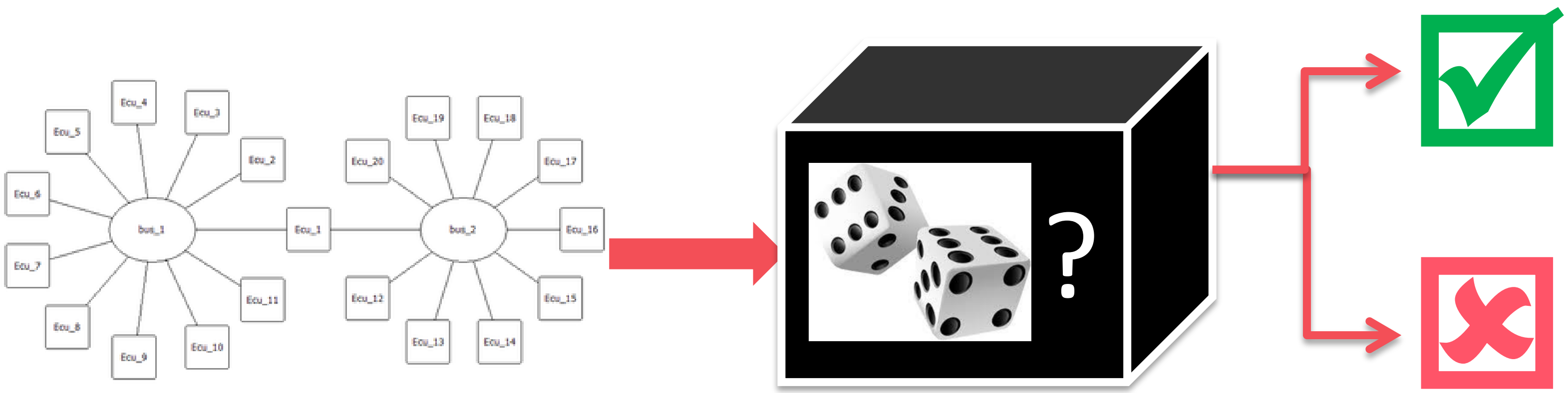


3 **Execute**

bare metal or hosted by an OS - prototypes or real systems

Freely available from www.designcps.com

How do we know simulation models are correct?!



What do we have at hand ?

- ✓ Are the models described ? Usually no
- ✓ Is source code available? No

} Black-box tools

- ✓ Complexity of the models and implementations? High – Domain experts typically take many months to master a new technology!
- ✓ Do we have qualification ? No
- ✓ Are there public benchmarks on which validate the results? No
- ✓ Limited number of end-users and cost-pressure ? Yes
- ✓ Can we prove the correctness of the simulation results ? No

Best practice : several techniques and several tools for cross-validation

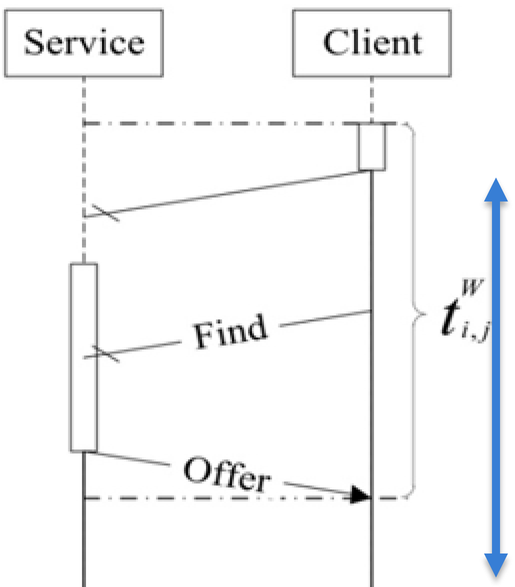
Examples of cross-validation

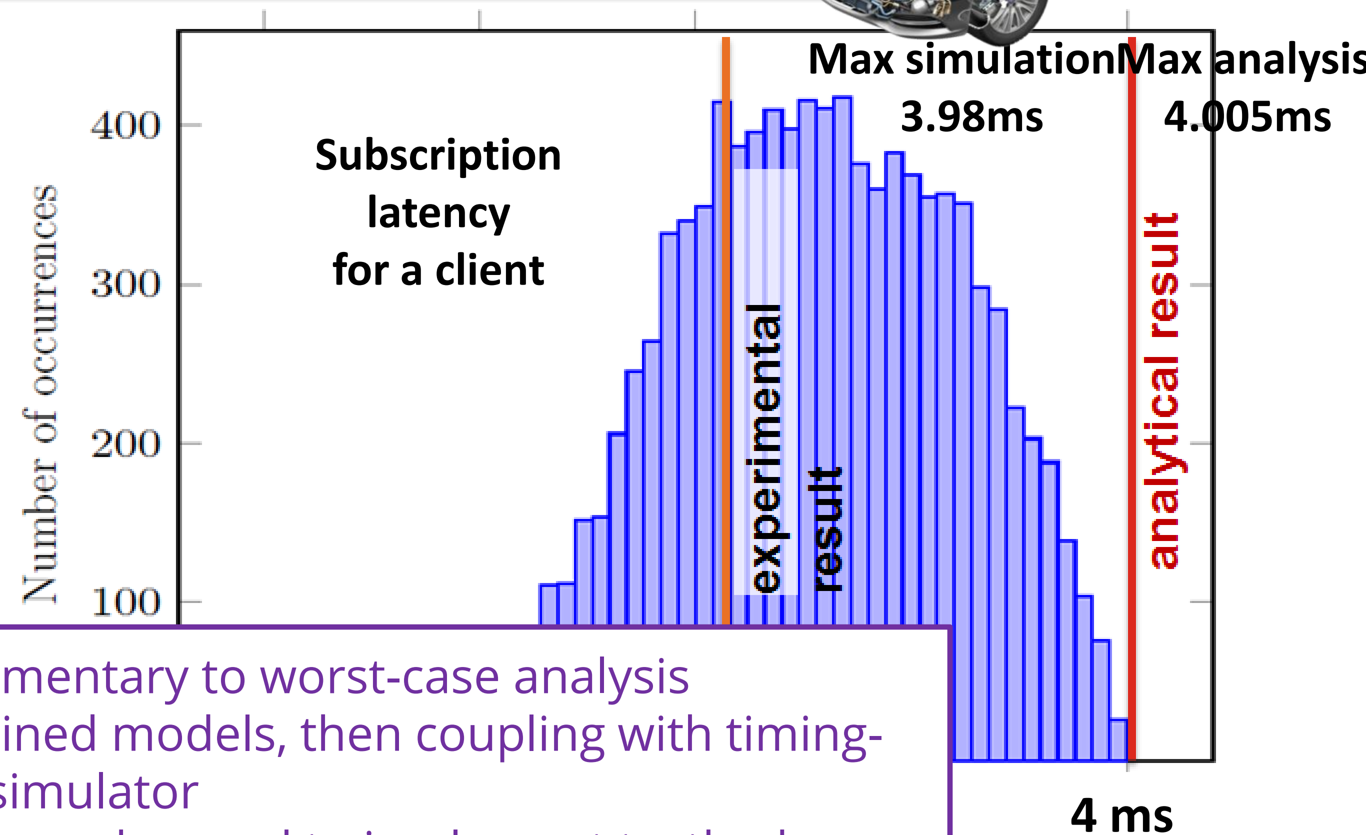
- ✓ **Comparing different simulation models:** e.g, in-house vs commercial, coarse-grained vs fine-grained
- ✓ **Comparing simulation against analytic results:** e.g., upper-bound and lower-bounds analysis
- ✓ **Validating a simulator using real communication/execution traces:** e.g., comparing inter-arrival times distributions
- ✓ **Re-simulating worst-case situation from mathematical analysis**
- ✓ …

Our experience: for complex systems, validating timing accurate simulation models is much easier than mathematical models

Illustration: Some/IP middleware [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 can be used to implement testbeds

Simulation for .. safety-critical systems ?!

Our view: if system can be made robust to rare (quantified) faults such as 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 – see [1]

- ✓ **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 o
- ✓ Q2:
- ✓ Q3:

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

Simulation length choice

Period: 80 ms — Robust quantile: Q5

Independent Runs: 1

Required length: d 22 h 13 m s ms µs

Robustness of quantiles

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	-	-

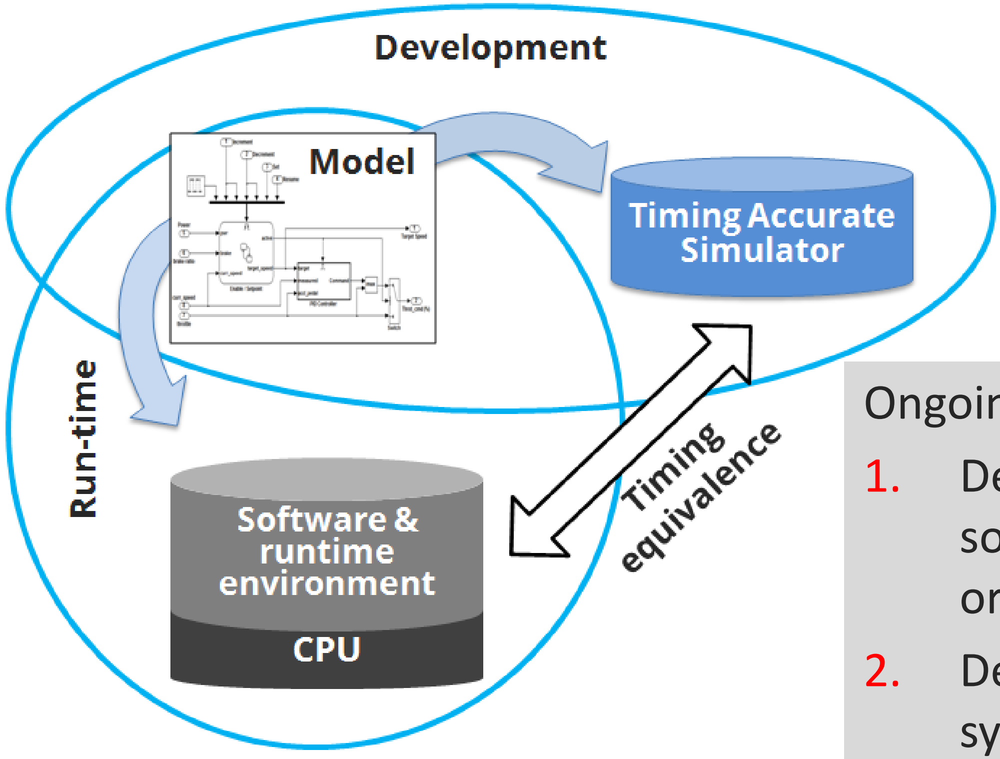
Min	Average	Q2	Q3	Q4	Q5	Q6	Max	Bound
						477 ms	0,477 ms	0,550 ms
						719 ms	0,719 ms	0,830 ms
						925 ms	0,925 ms	1,074 ms
						167 ms	1,167 ms	1,354 ms
						943 ms	0,943 ms	1,092 ms
						185 ms	1,185 ms	1,372 ms
						427 ms	1,427 ms	1,652 ms
						669 ms	1,669 ms	1,932 ms
0,148 ms	0,217 ms	0,773 ms	1,079 ms	1,275 ms	1,328 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

[RTaW-pegase screenshot]

Industry trend: verification by simulation implemented as a push-button feature in the design flow with all the complexity hidden from the user - domain expert only called on in case performance requirements are not met.

Ahead of us #1 : timing-Augmented Model Driven Development

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

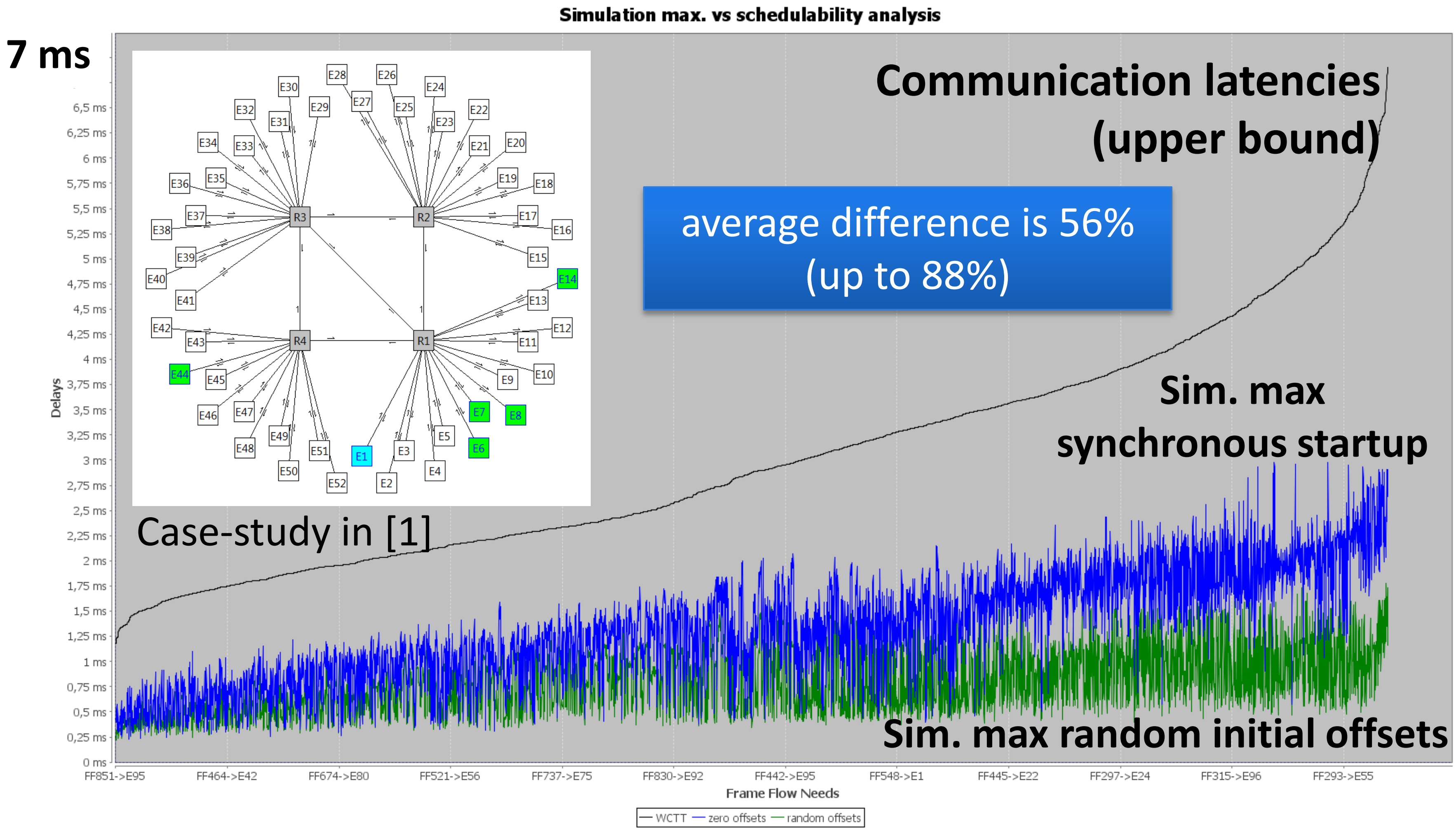


Solution: injecting delays in the simulation - but how to do that early stage without knowledge of complete configuration ?

Ongoing work:

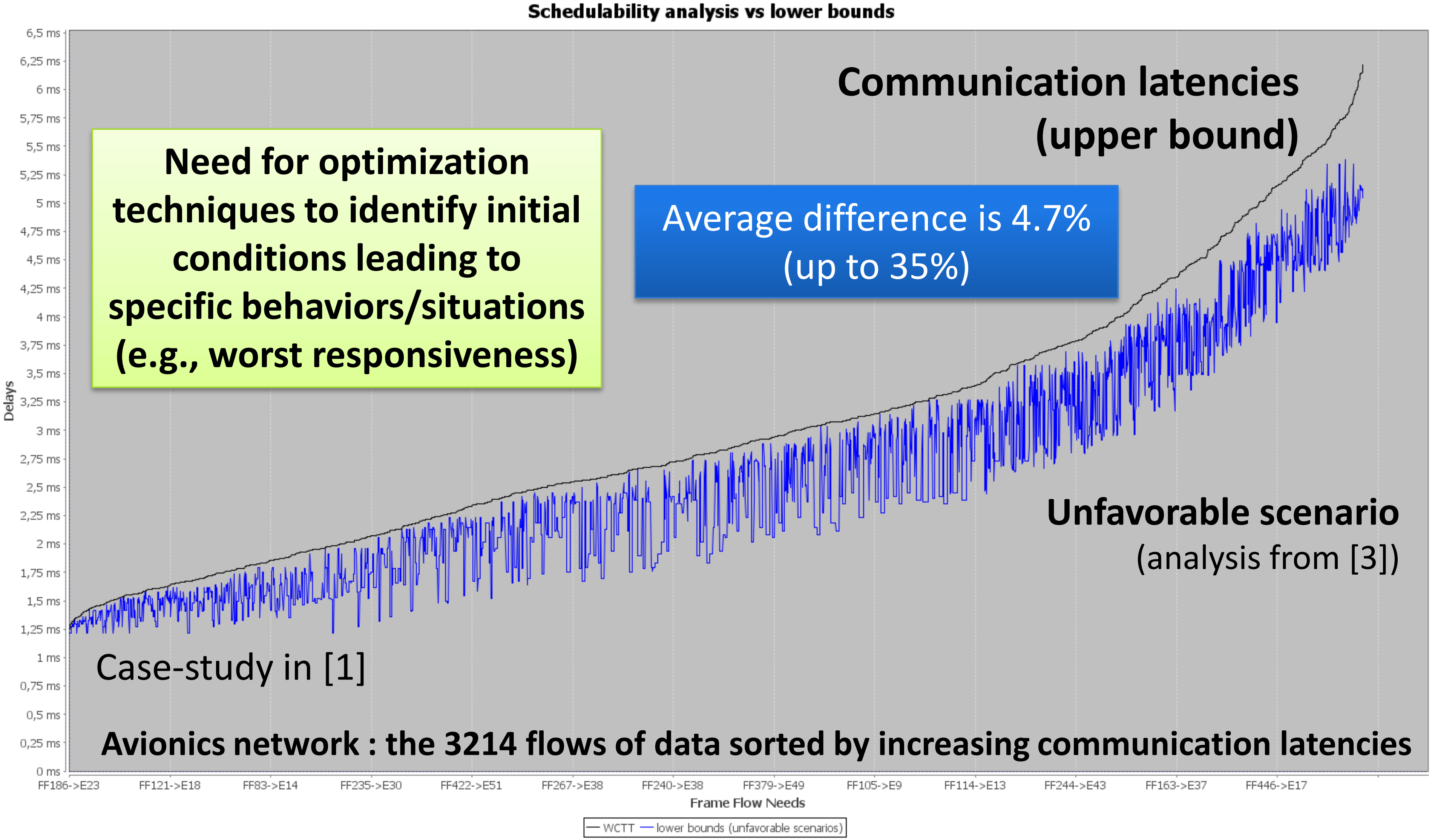
1. Designer defines timing-acceptable solution in terms of significant events: order & quantified relationships btw them
2. Derive QoS needed from the runtime systems: CPU, comm. latencies
3. Resource reservation & QoS ensured at run-time

Ahead of us #2 : finding initial conditions leading to degraded performances → worst-case oriented simulation



Avionics network : the 3214 flows of data sorted by increasing communication latencies

Ahead of us #2 : simulation is unable to find pessimistic situations .. unlike lower bound analysis



Key takeaways

- ✓ **Complex mathematical models is a dead-end for systems not conceived with analyzability as a requirement** → they cannot catch up with the complexity - see [1]
- ✓ **Simulation is effective for critical systems that can tolerate faults with a *controlled* risk** → best resource usage
  - **Need for proper methodology**
  - **Cross-validation is a must-have**
  - **Models and their assumptions should be questioned by end-users**
- ✓ **Today: high-performance timing-accurate simulation of complete heterogeneous embedded architectures**
- ✓ **Ahead of us: system-level simulation with functional behavior within a Model-Driven Engineering flow**

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] 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.

[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.

[9] F. Boniol and V. Wiels, "Landing gear system", case –study presented at ABZ2014, 2014.

[10] AUTOSAR, "Specification of Timing Extensions", Release 4.0 Rev 2, 2010.

[11] M. Tatar, "Inside an Engine ECU - a view you've not seen before", Linkedin Pulse, 2016.