Automotive multi-core trends in Germany

Embedded Technology 2016, Japan
Multi-/many-core summit

Peter Gliwa
CEO GLIWA GmbH

Version 2

Contents

- Introduction
- Multi-core in automotive projects
- Timing requirements in OEM's requirements specifications
- State-of-the-art multi-core timing analysis
- Upcoming Standard: ARTI (AUTOSAR Real-Time Interface)
- Summary

Introduction

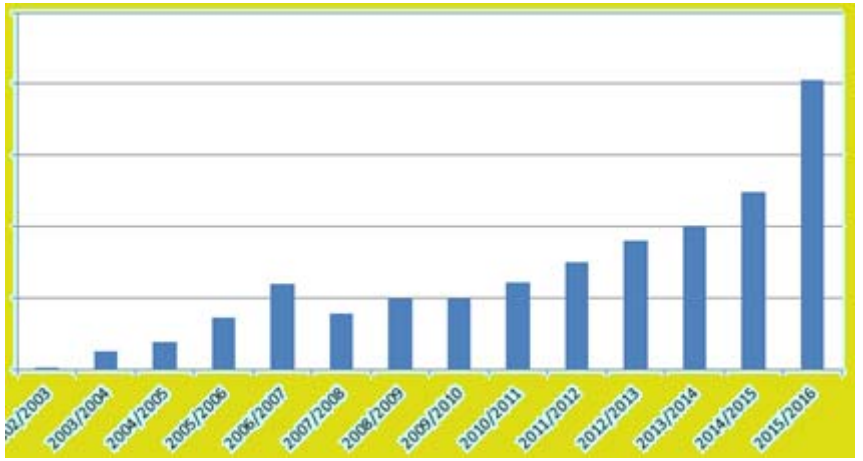
Gliwa GmbH – company introduction

- Timing analysis and embedded software expertise since 2003
  - embedded timing secured in **hundreds of mass-production projects**
  - located near **Munich** in Weilheim i.OB., **Germany**
  - 26+ employees mostly **engineers**
- **Stack Analysis** combining static and dynamic methods
- Distributor for Japan: Solution House GAIO TECHNOLOGY

www.gliwa.com

Latest figures/news

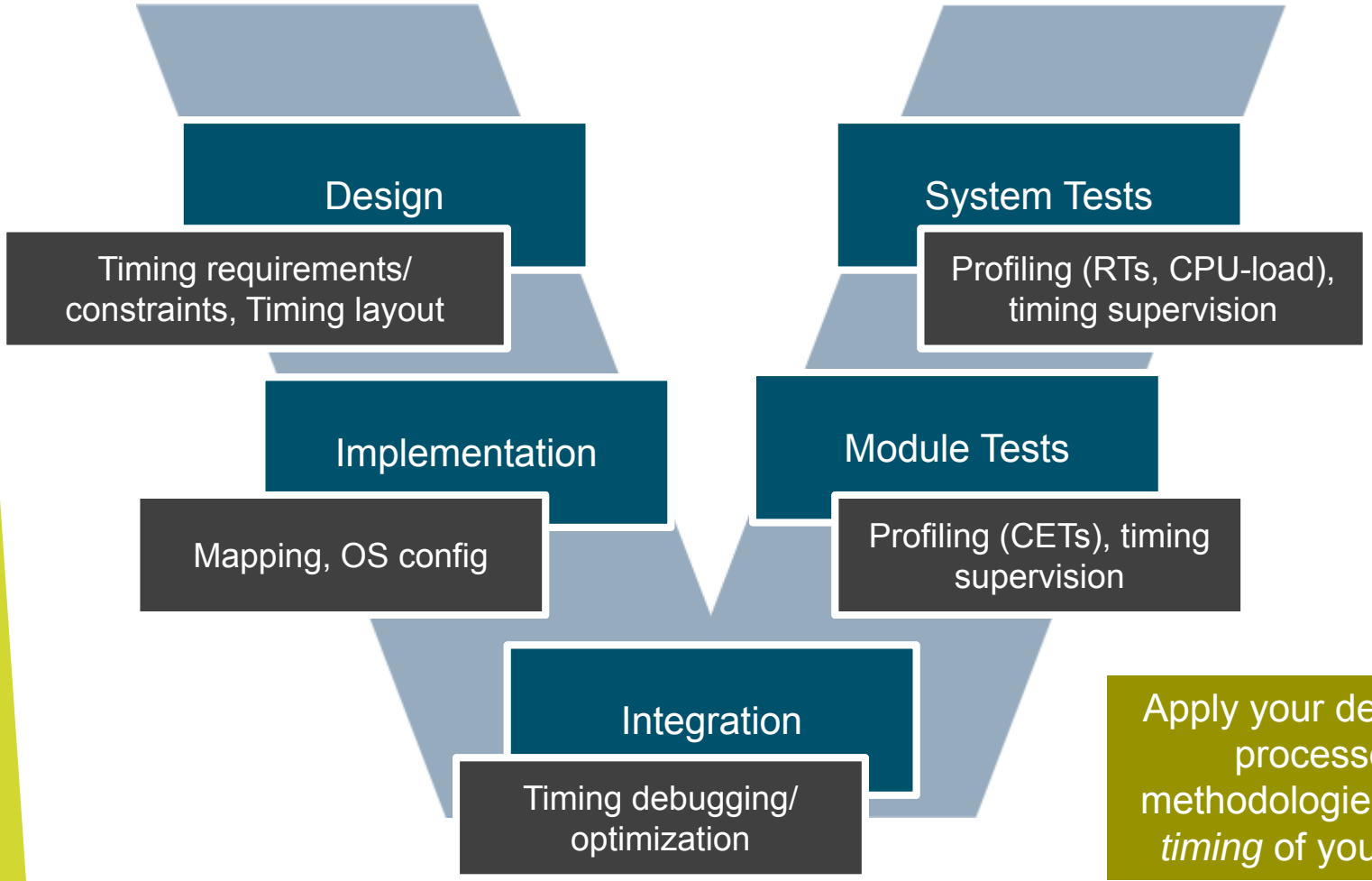
- T1 is the de facto standard for tracing in the German automotive market
- T1 is required by several OEMs
- Constant growth over the past years (mostly >20%pa)
- GLIWA Ltd. in York (UK) founded in 2015



Yearly turnover

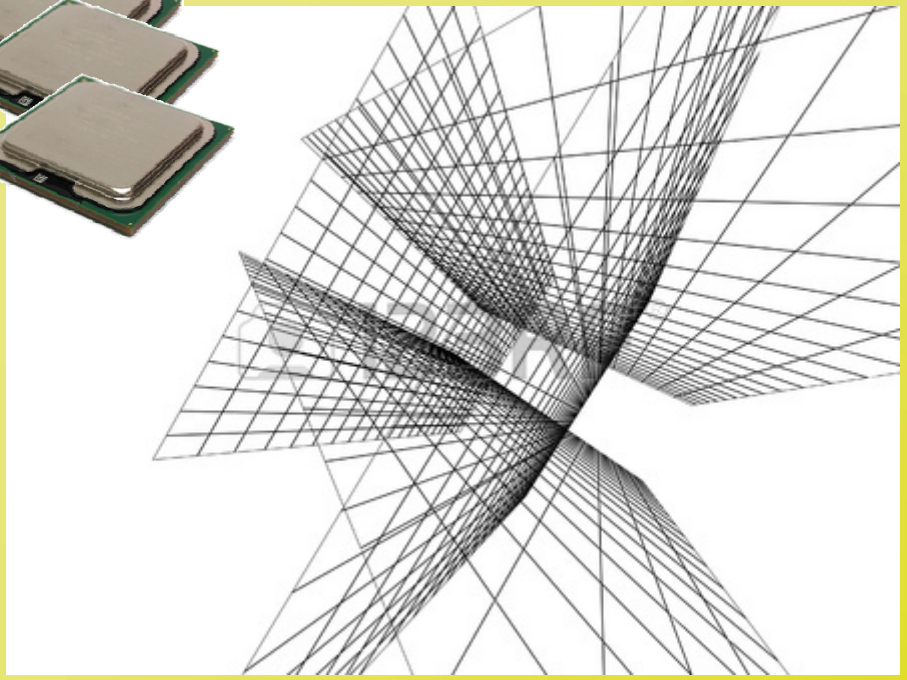
- GLIWA engineering founded March, 2016

Timing: a new species?



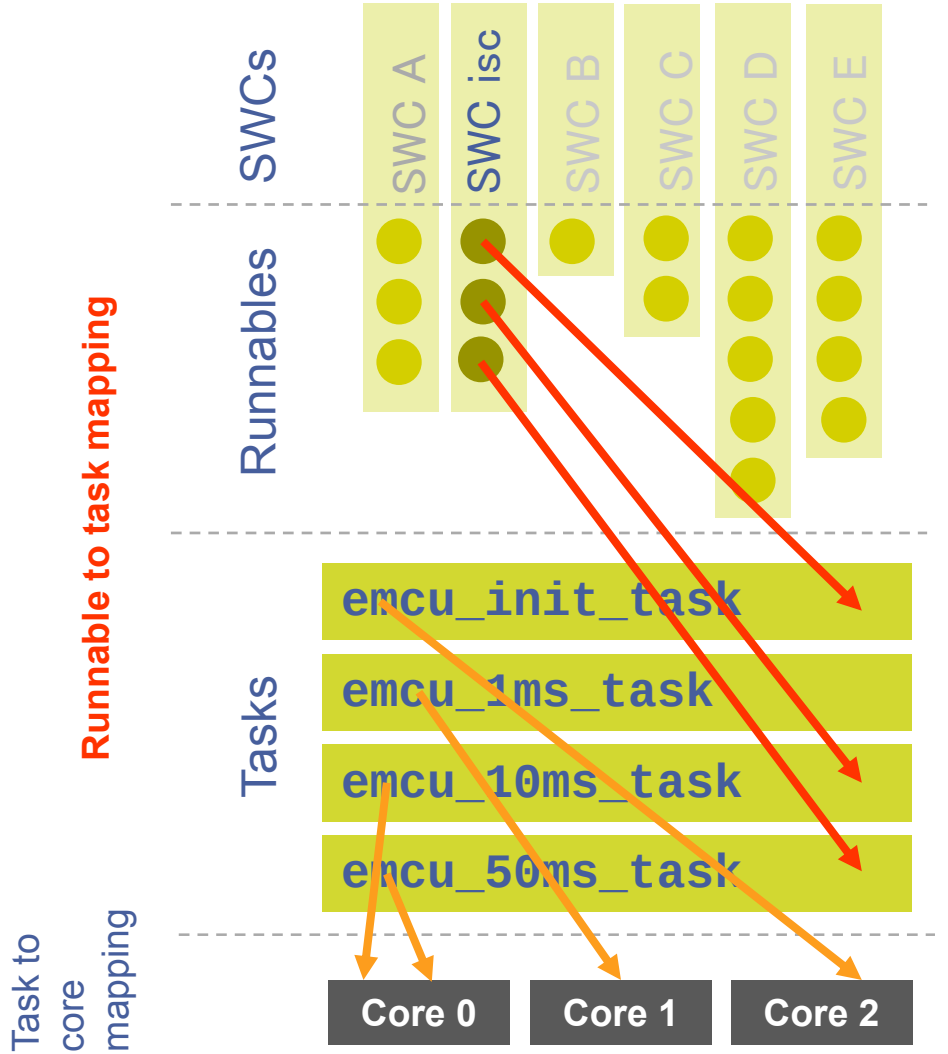
Apply your develeopment processes and methodologies also to the *timing* of your software!

Multi-core in automovie projects

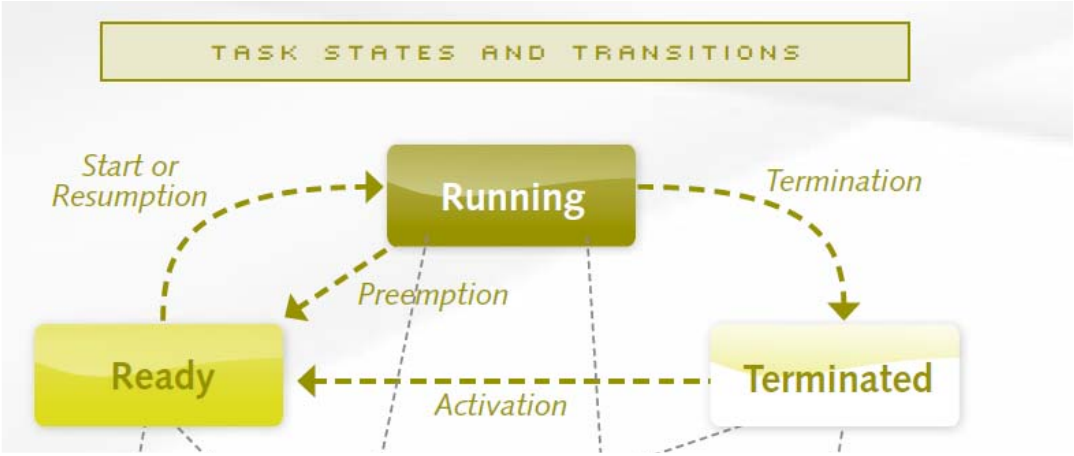


AUTOSAR scheduling: definition of terms, example

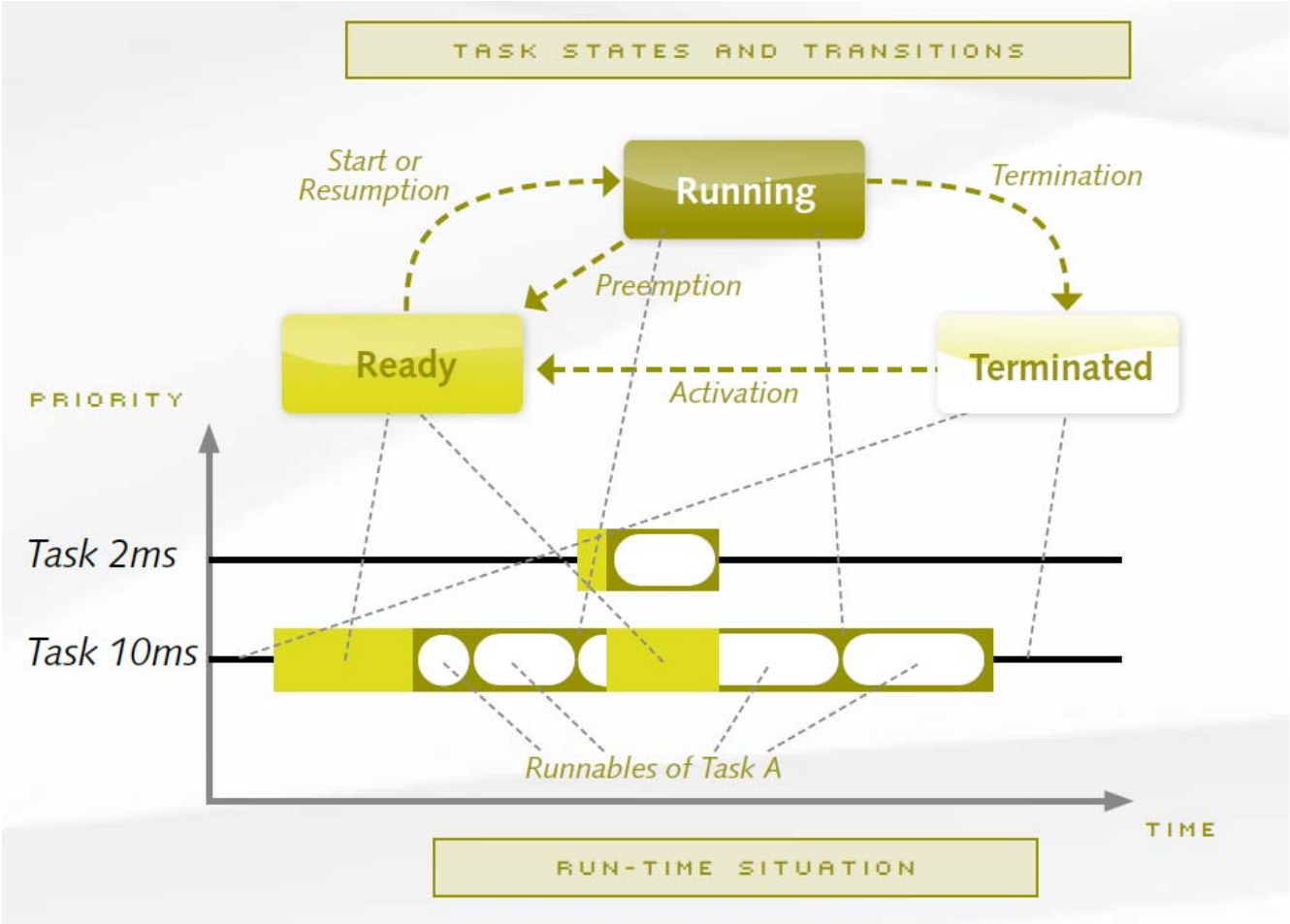
- A **software component** (SWC) “idle speed control” of an engine management ECU is coded in three **runnables**:
  - IdleSpeedInit
  - IdleSpeed10ms
  - IdleSpeed50ms
- As part of the RTOS configuration, these get mapped to three different **tasks** which they share with many other runnables from other SWCs.
- For multi-core processors, tasks get also mapped to a certain **core**.



AUTOSAR scheduling: task states

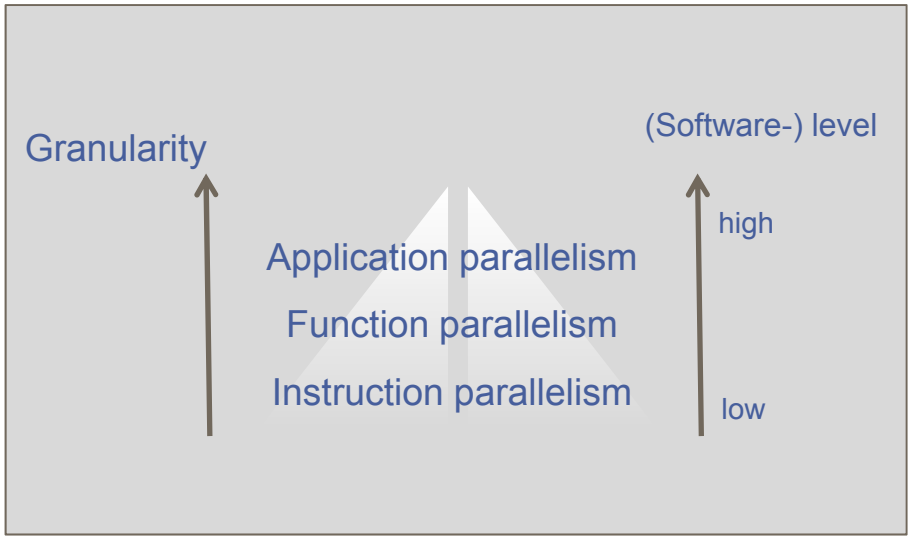


AUTOSAR scheduling: run-time situation



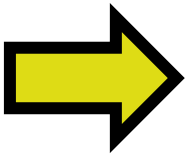
Multi-core: different kinds of parallelism

- Application parallelism
  - Each application runs on one core only.
  - Think of independent threads under a desktop OS
- Function parallelism
  - Executing portions of functional closely related code in parallel
  - Think of a sort algorithm
- Instruction parallelism
  - Processor cores have pipelines which process instructions in parallel.



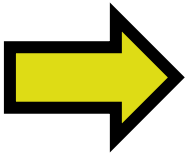
Multi-core: different kinds of parallelism

- Application parallelism
  - Each application runs on one core only.
  - Application ≠ thread



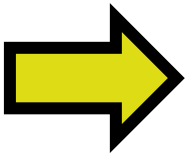
- **Easy:** take two single-core ECUs and assign each SW to the core of a multi-core ECU

- Function parallelism
  - Executing portions of functional related code in parallel



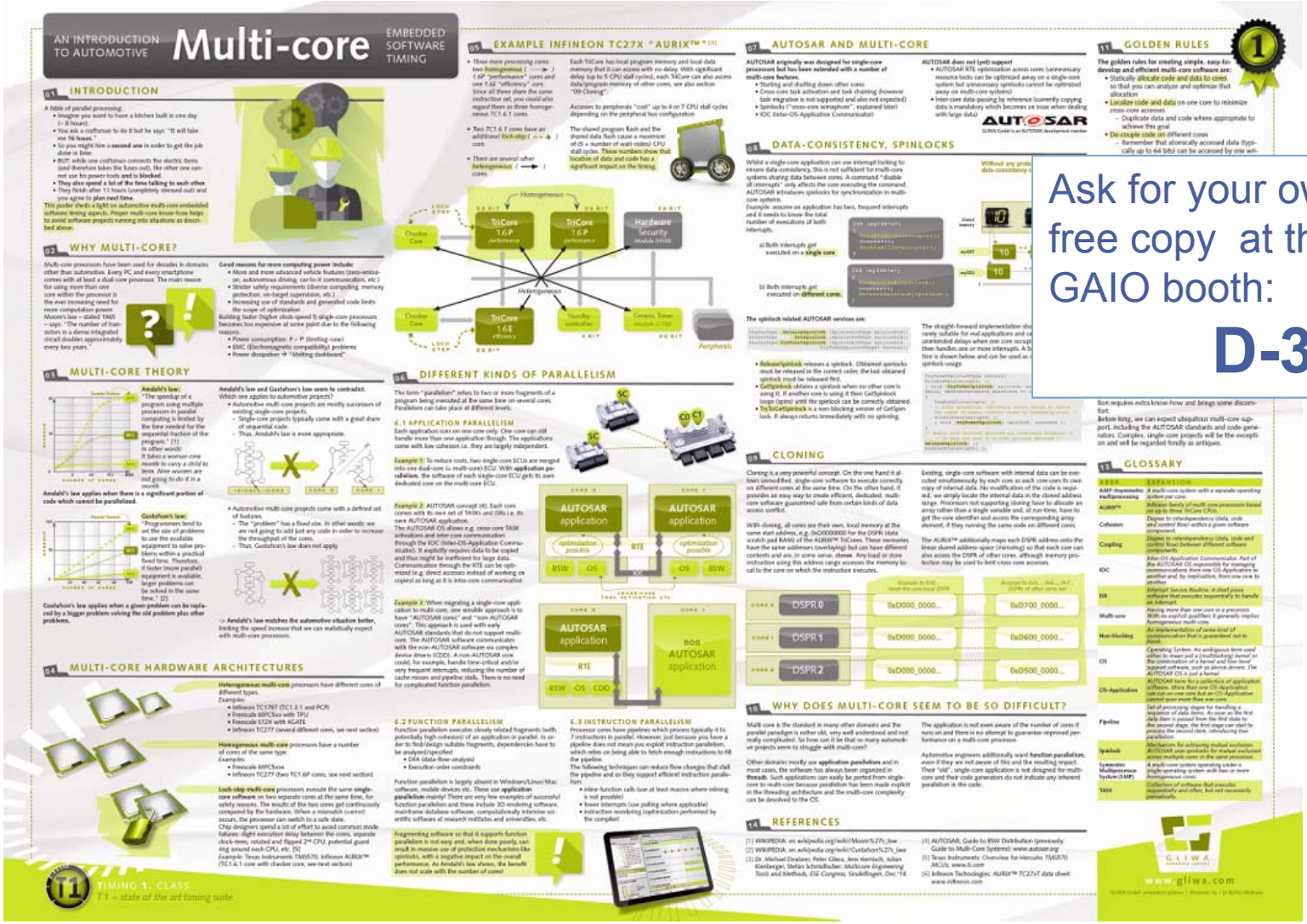
- **Extremely difficult:** automotive SW is typically not designed to be parallelized

- Instruction parallelism
  - Processor cores have pipelines which process instruction in parallel.



- **Mostly easy:** follow some design rules to exploit your pipeline

The Multi-core Poster – Multi-core on one sheet of paper



Ask for your own free copy at the GAIO booth:

D-34

Timing requirements

today and tomorrow

Requirements specification documents

- A good ECU's requirements specification is the foundation for sound and safe timing.

- Requirements specifications should addresses two topics related to timing:
  - A list of timing requirements (as far as they are known)
  - Requirements regarding the environment, methodologies and tools

timing requirements

methodologies & tools requirements

Timing requirements

timing requirements

- Known timing requirements, e.g.
  - start-up times (presence on bus)
  - sub event chains (as part of end-to-end constraint)
  - execution orders
- Budgets
  - overall CPU-load
  - CETs (also for containers holding OEM code!)
- ISO 26262
  - requires “freedom of interference”
  - in other words: correct and secured timing

Methodology requirements

methodologies & tools requirements

- Define a process/infrastructure for maintaining timing requirements → handle upcoming timing requirements
- Select appropriate timing analysis techniques, e.g.
  - scheduling simulation for the early project phase and
  - traing/measurement for the late phase
- Specify minimum requirements regarding timing requirements *verification*
  - Specify when/how often timing requirements get verified
  - Specify the environment (in-car, lab, simulation, static analysis, …)
  - Specify the conditions (scenarios, test coverage, …)
- Specify how timing verifications and timing properties are documented.

Tool requirements

- Typically, the OEM does not require the supplier to use a certain tool.
  → However, in some cases this might make sense.
- Specify who has the tools available.
  → In-house solutions do not help much in a shared SW development.
- Specify which exchange formats and export formats have to be supported.

methodologies & tools requirements

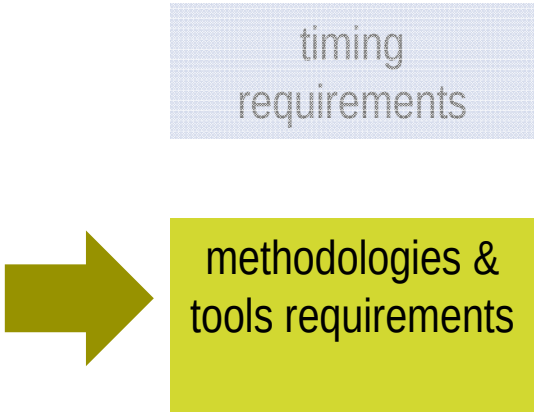
Example 1: OEM wanted to measure, supplier provided in-house tool. Interfaces were often subject to changes, no manuals, no dedicated support
→ usage at OEM very difficult

Example 2: OEM forgot to specify they want to able to measure. So they did not get access to the existing supplier internal tools.

Templates for requirements specifications

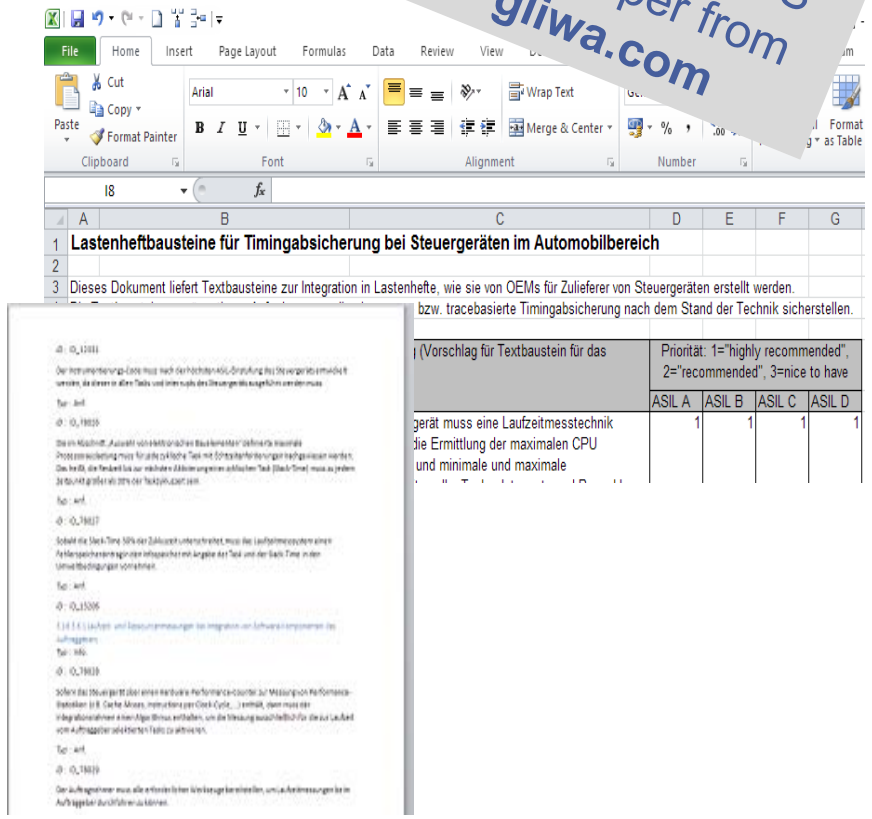
- In 2009, BMW started collecting generic text blocks for requirements specifications
  - Ensure that the timing topic is addressed properly in future projects.
  - Reuse approaches which were successful in the past.
  - Efficiency: avoid that authors of requirements specifications have to ‘research’ timing related aspects again and again.

- The result: text templates for requirements specifications.
  - Timing requirements are mostly project-specific
  - Thus, templates focus on methodologies and tools.

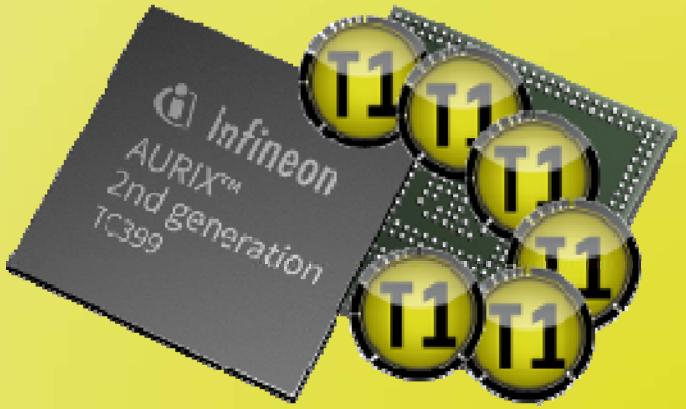


Templates for requirements specifications

- Excel table with text templates
  - including recommendation according to ASIL level
- Word document with templates
- Some big OEMs follow this approach already. More and more follow…



State-of-the-art multi-core timing analysis



Instrumentation-based tracing: T1

T1-HOST-SW

PC based SW tool for visualization, analysis and configuration

The T1-TARGET-SW is certified according to **ISO 26262 ASIL-D**

T1-TARGET-SW

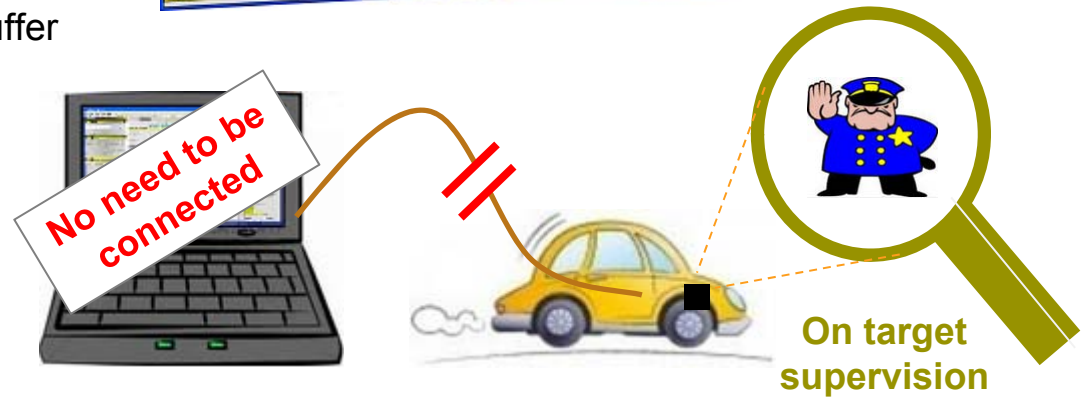
Embedded software component which traces, analyses and supervises at run-time

Instrumentation based tracing – “online” *and* “offline”

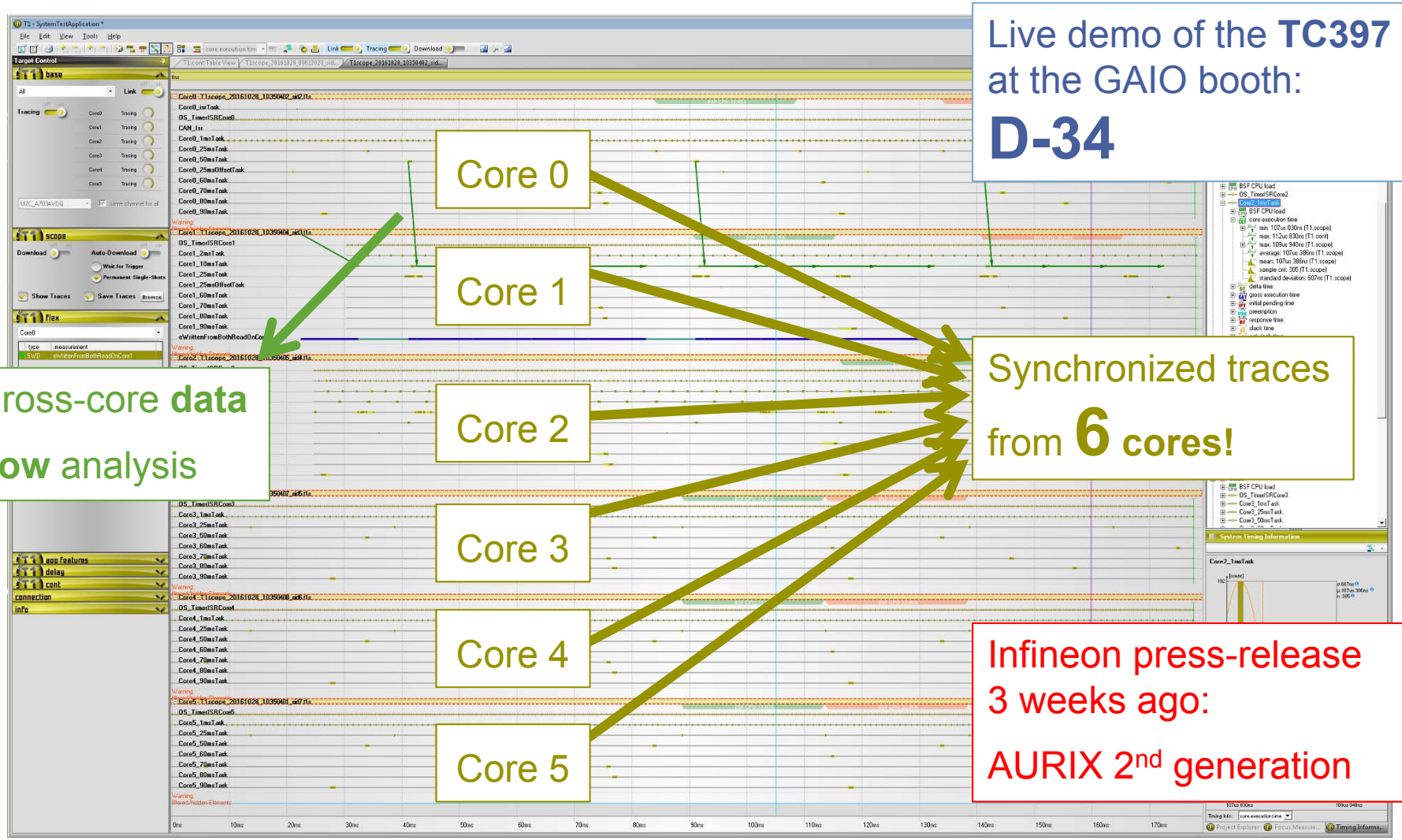
- The trace data can be analyzed
  - Off-target (including visualization)
  - On-target (continuous “on-the-fly analysis”)

- On-target analysis allows
  - not only measurements but supervision: compare measured parameters with pre-defined limits
    - Continuously
    - In-car without PC connected
    - Callbacks allow e.g. error buffer entries
  - storing results in non-volatile memory → collect results from billions of executions

Visit us at the GAIO booth:
D-34



Latest news: synced traces fro 6 cores



Live demo of the **TC397** at the GAIO booth:

D-34

Cross-core **data flow** analysis

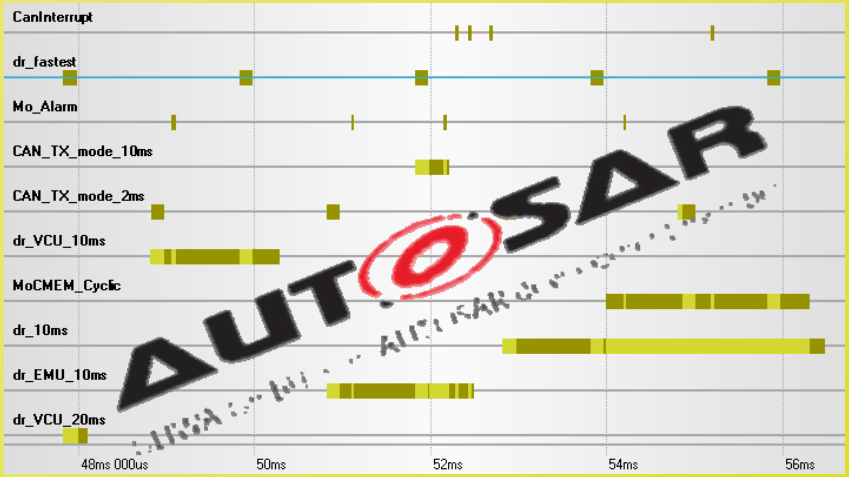
Synchronized traces from **6 cores!**

Infineon press-release 3 weeks ago:

AURIX 2nd generation

ARTI

AUTOSAR Real-Time Interface



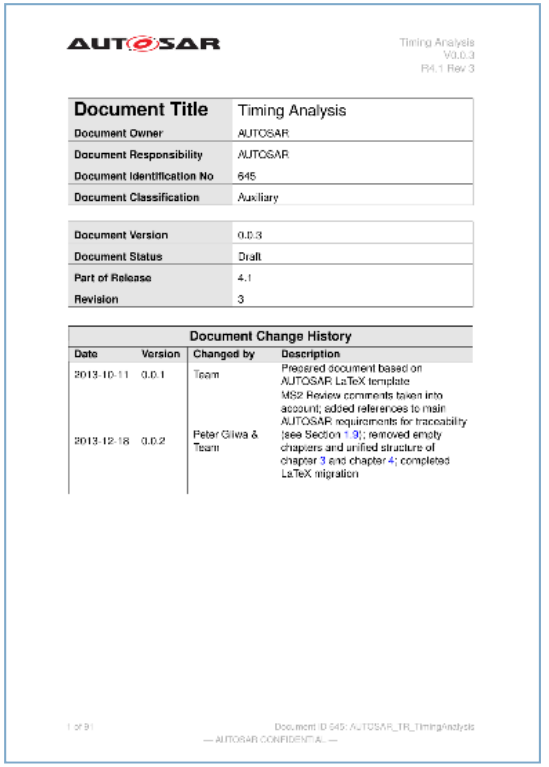
AUTOSAR and timing

- **Timing Extensions**
  ”TIMEX”; since AUTOSAR 4.0
  → Allow specification of timing requirements

- **Timing Analysis**
  First released with 4.1.3
  → Use-cases based guide to timing

- **AUTOSAR OS**
  Contains timing protection mechanisms
  → As of 4.2.2: Execution-, Locking- and Inter-Arrival Time Protection

- **ARTI** (AUTOSAR Run-Time Interface)
  not a standard yet (probably in 2017)
  → more details later



AUTOSAR Timing Analysis V0.0.3 R4.1 Rev 3

Document Title	Timing Analysis
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	645
Document Classification	Auxiliary
Document Version	0.0.3
Document Status	Draft
Part of Release	4.1
Revision	3

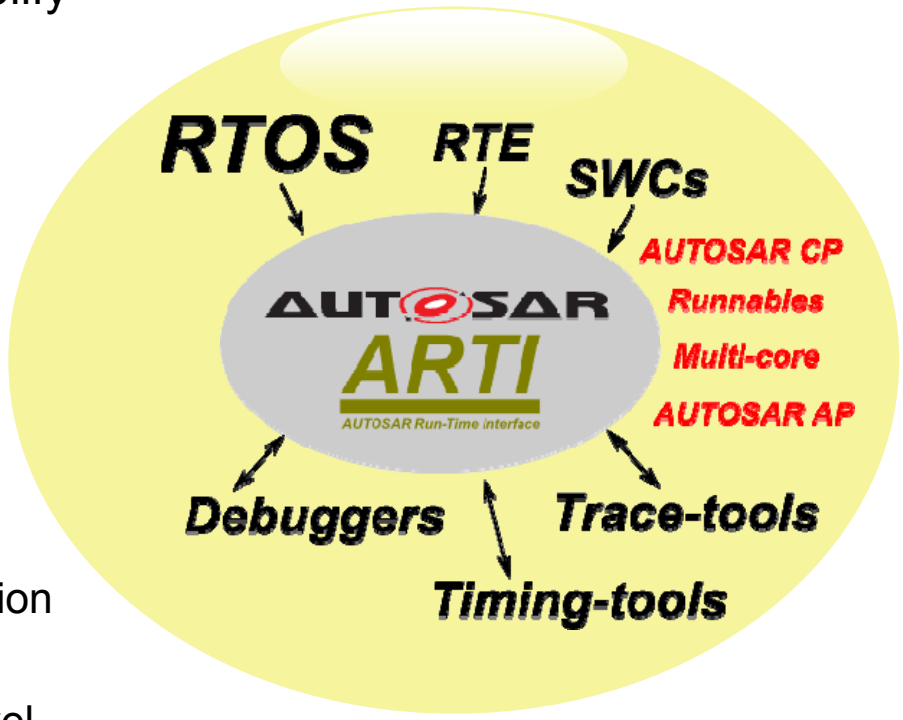
Document Change History

Date	Version	Changed by	Description
2013-10-11	0.0.1	Team	Prepared document based on AUTOSAR LaTeX template
2013-12-18	0.0.2	Peter Gliwa & Team	MS2 Review comments taken into account; added references to main AUTOSAR requirements for traceability (see Section 1.3); removed empty chapters and unified structure of chapter 3 and chapter 4; completed LaTeX migration

Timing Analysis document

ARTI goals

- Generally, ARTI shall support and simplify the following:
  - Debugging
  - Tracing
  - Timing Measurement / Profiling
- ARTI shall support
  - Multi-core
  - Runnables
  - Instrumentation-based tracing and measurement solutions
  - The actual AUTOSAR-OS implementation
  - TIMEX
  - debugging and tracing beyond ECU level, e.g. end-to-end timing taking several ECUs and buses into account
  - AP (adaptive platform)



ARTI: who is behind it?

AUTOSAR/ OS vendors

ARC CORE

ETAS

GLIWA

VECTOR

NXP

Timing Tool vendors

INCHRON
THINK REAL-TIME

GLIWA

Timing Architects

Debug/Trace Tool vendors

iSYSTEM

LAUTERBACH
DEVELOPMENT TOOLS

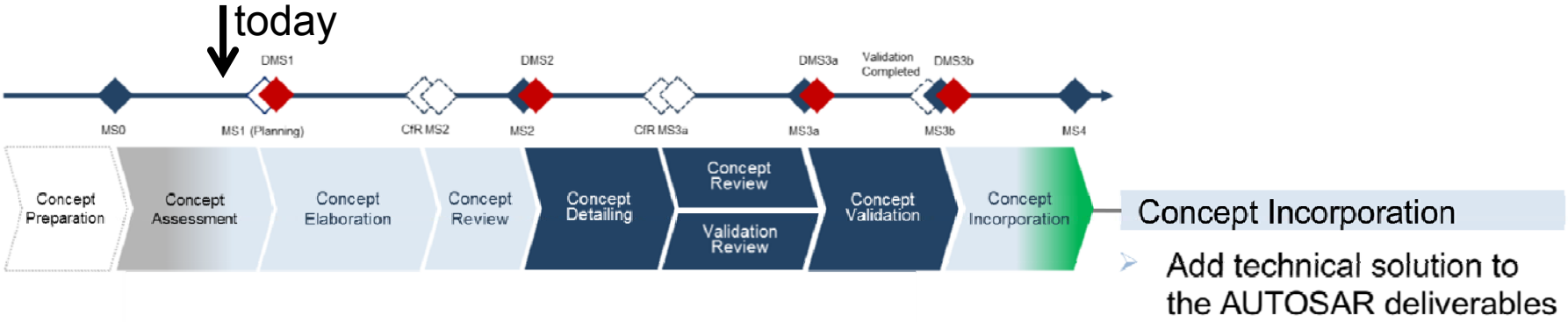
Users, AUTOSAR experts

Continental

LieberLieber

ARTI: current status/schedule

- Concept owner: Peter Gliwa <peter.gliwa@gliwa.com>
- MS0 passed 2016-09-07
- Further planning:
  - MS1 (WP Assessments completed): 2016-11-25
  - MS2 (review readiness): 2017-02-15
  - MS3a (Call for review): 2017-09-15
  - **MS3b (Validation completed): 2017-11-01**



Conclusion

Conclusion

- More detailed timing requirements are necessary for building safe and reliable systems.
- Well engineered requirements regarding methodologies & tools are necessary for efficient SW development.
- Tracing gets your timing on the safe side.
- Developing multi-core ECU SW without timing tools is far too costly and error prone.

peter.gliwa@gliwa.com

Thank you