UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

Engineering of Reliable Software Systems

Compliance of functional and non-functional requirements of embedded systems by model-driven software engineering

Dipl.-Ing. Harald Hauff
Prof. Dr. Hermann de Meer

Computer Networks & Communications
Prof. Hermann de Meer

ISL Institute of IT-Security and Security Law

Arrangement

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- Some words to the past and the present status of model - driven SW engineering
- Overview of model-driven SW engineering (steps)
- Gaps in model-driven approaches with respect to functional safety
  - System behaviour
  - Safety integrity
- Two selected topics and an approach to a solution
- Perspective
- Conclusion

Status of model-driven SWE

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- In the past:
  - A lot of paradigm changes in
    - Specification /modeling languages (SADT, UML, ER, SDL … formal methods like Z, CSP, VSE, …)
    - Programming languages (object code, Assembler, imperative PL, object-oriented languages like C++, ADA, …)
    - Function-oriented (not safety-oriented)
- Present used methods/tools for embedded systems:
  - Mainly UML, tool specific languages
  - Implementation language „C“ (C++), „Ada“
  - Tools: e.g. MATLAB® Simulink® (MathWorks), SCADE (Esterel)
- Results: furthermore blue screens

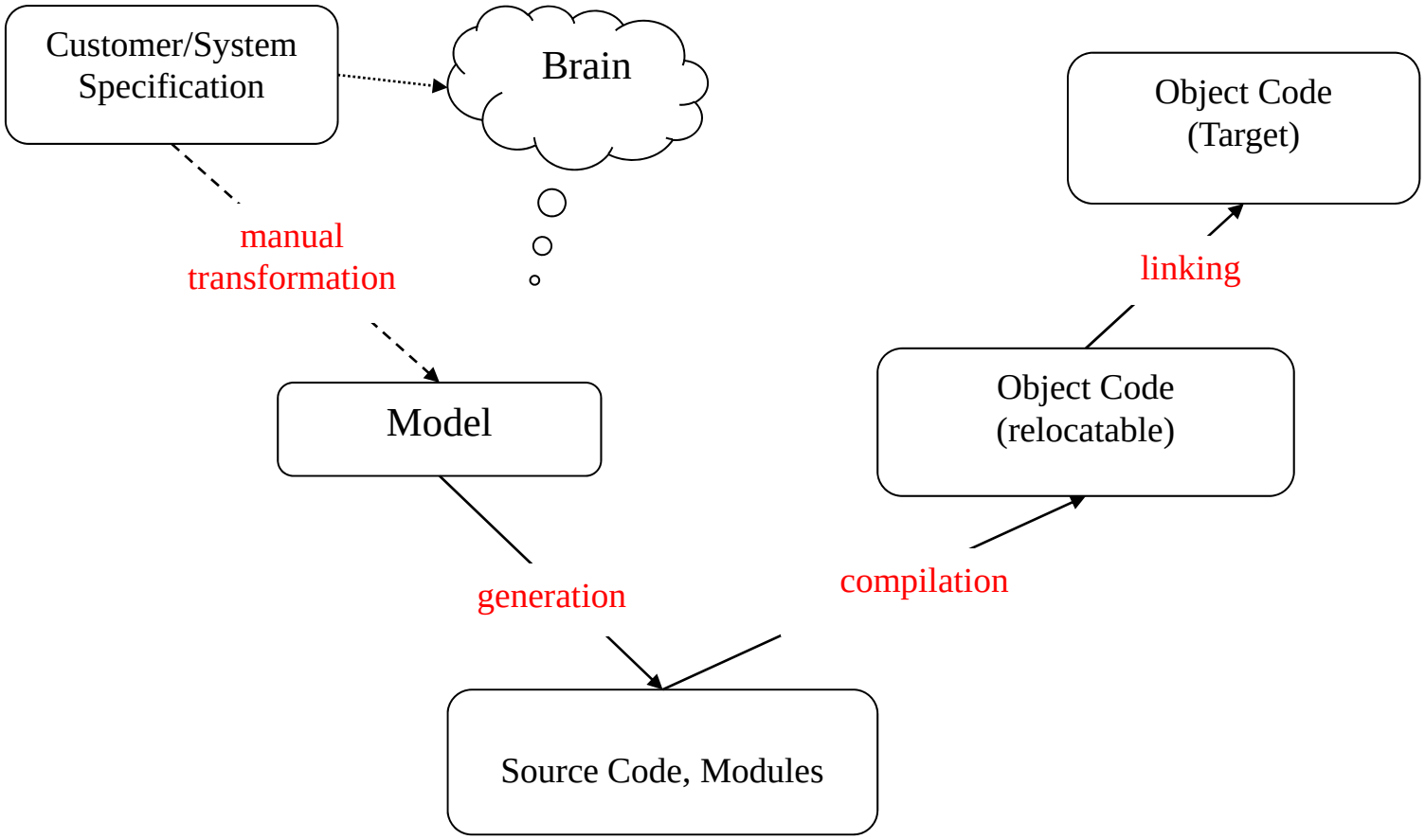
Status of model-driven SWE

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- Is model-driven engineering suitable to develop reliable systems according to functional safety?
  - (YES)
  - But additional progress is required for real time behavior, maintaining semantics by generating code and safety integrity

- Questions about the usage of methods and tools
  - Is the method suitable to describe the behavior of the real system
  - Is the programming language suitable to implement the system (multitasking, communicating processes, real time, …)

Model-driven SWE (Steps down)



Gaps in model-driven SWE

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- Gaps about system behavior, e.g.
  - Confirmation of real-time behavior
    - Some approaches available for sequential processes or deterministic systems (e.g. SCADE)
    - Difficulties for concurrent/multithreaded/asynchronous tasks
  - Avoidance of unexpected „trips to fail“
    - Semantic changes by generation of source code or compilation

Gaps in model-driven SWE

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

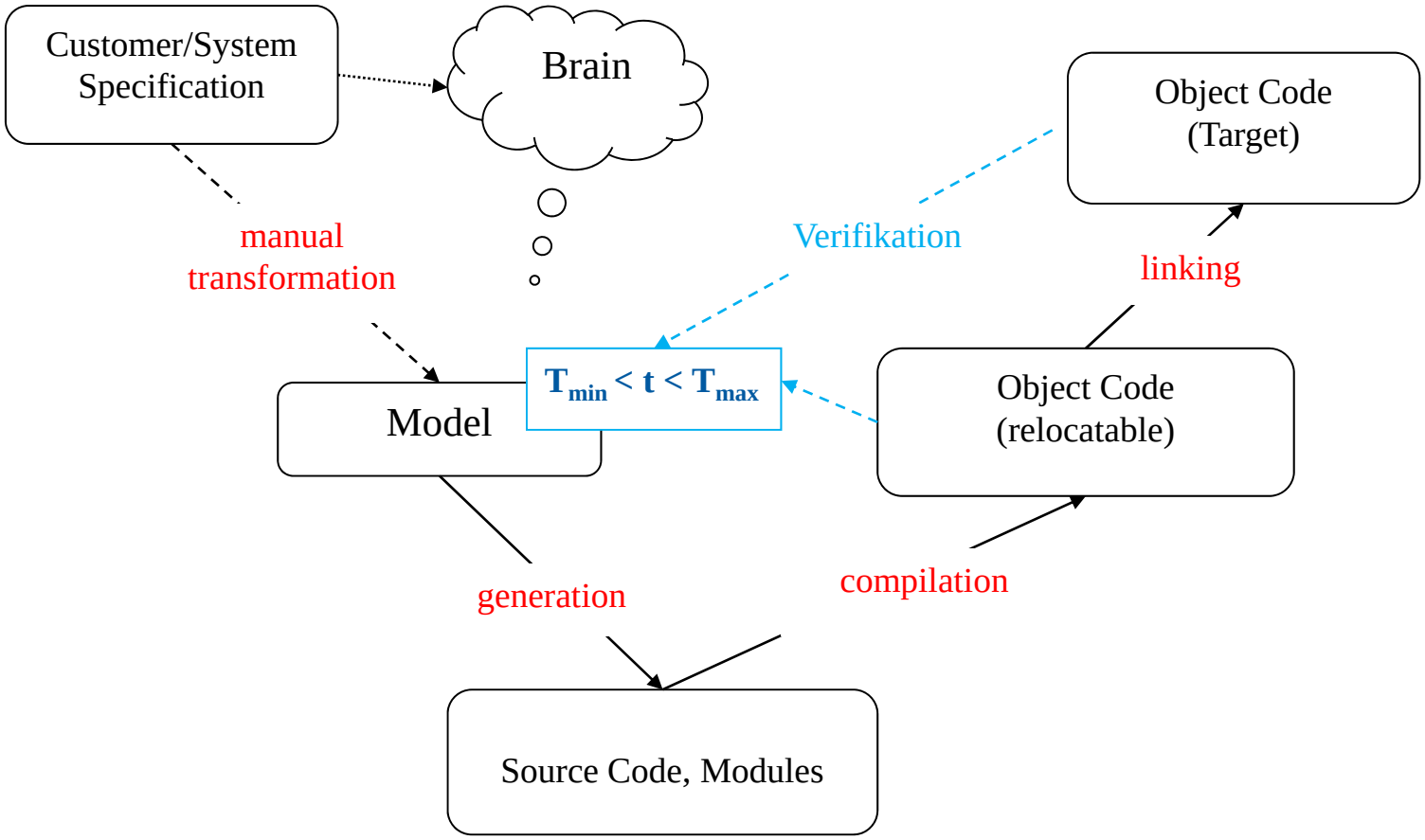
- Gaps about integrity requirements (at runtime)
  - Detection of static faults up to SIL 2 (according to IEC 61508)
    - E.g. CPU, RAM, ROM test
    - Depending on used resources of the system
  - Detection of transient faults at SIL 3 and higher
    - E.g. Diversity (types: data storage, time, operation, code)
    - Types of diversity like:
      - Data storage (redundant, 1er-complement)
      - Time diversity (running algorithm twice before decision making)
      - Operator diversity (e.g. shift and addition instead multiplication)
      - Code diversity

Real-time behavior

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- Additional attributes of models are needed to guarantee the correct execution time of a algorithm for a (e.g. closed-loop) control system
  - To avoid oscillation
  - Reaching the safe state within the process safety time
    - Fault detection time
    - Fault response time

Estimation of runtime

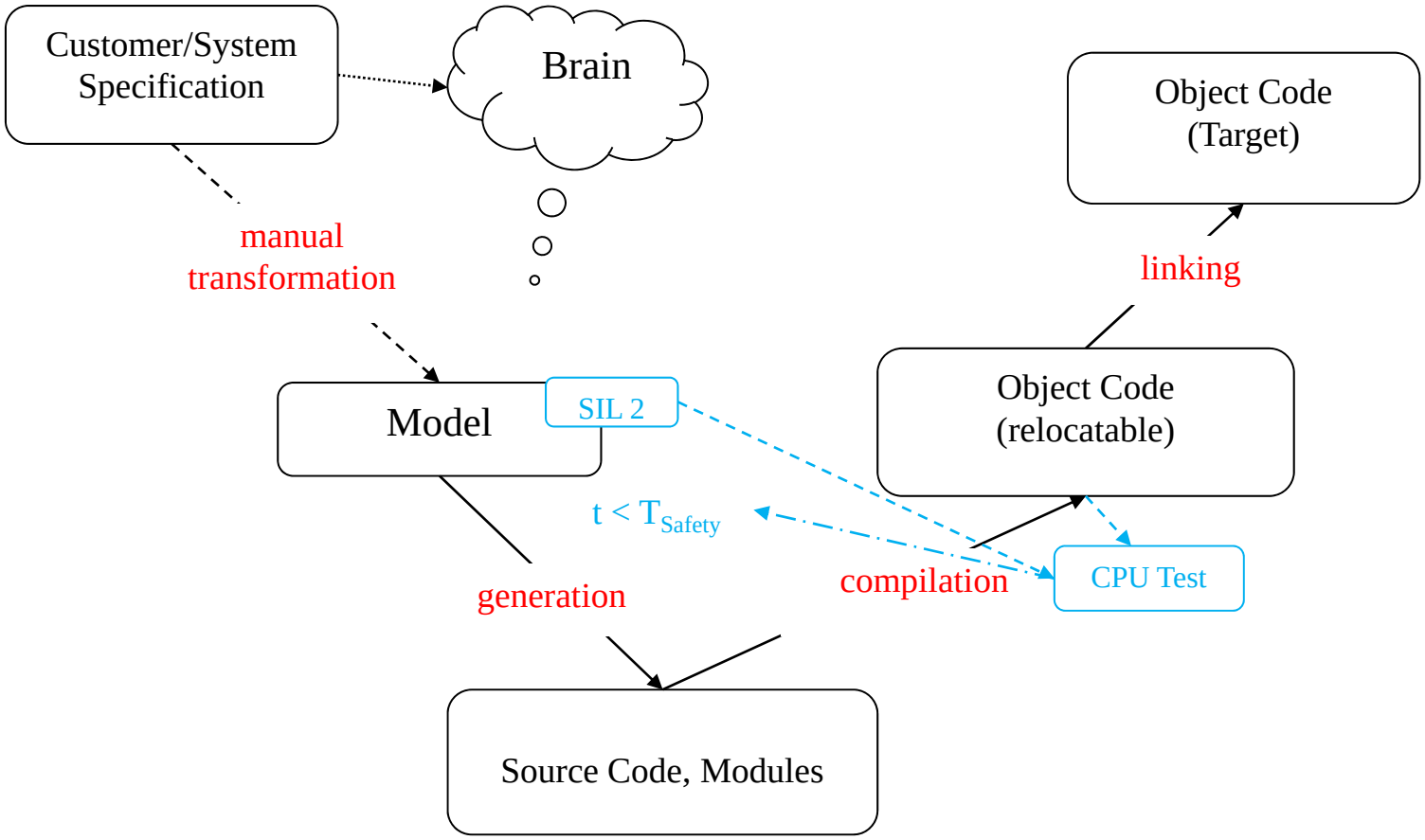


Real-time behavior

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- E.g. the tool SCADE from Esterel makes worst-case execution time (WCET) estimation for some microcontroller on the market
  - Limited to sequential and deterministic systems
  - Analysis of the generated object code depending on the target processor
- Missing:
  - For real-time systems and event-driven applications as well as for multi-tasking systems additional things must be considered, e.g.
    - Task changeover by the operating system
    - Frequency of interrupts
    - Nested interrupts
    - Abnormal interrupt behavior, …

Safety Integrity



Safety Integrity

- For SIL 2 systems a CPU test may be sufficient. It has to cover the used instructions and addressing modes
  - A CPU test can be automatically derived out of the generated object code (all used instructions and addressing modes)
  - The CPU test and the transition to safe state must be executed within the process safety time. The test interval and the fault response time must be considered!
- For SIL 3 systems a CPU test will be not sufficient. Integrity functions has to detect transient faults in operation
  - E.g. for data storage the code generator can use storing data twice with ones complement of data and address
  - E.g. for processing the code can be executed twice or diverse code can be used

Safety Integrity

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- Example for data integrity
  - Ones complement of data and memory address (memory map)

Var. a	Var. b	…					
					…	¬Var. b	¬Var. a

Safety Integrity (Development)

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- There are some approaches of tools in the market to use model-driven software engineering for verification and validation activities
  - Using the model to test software in the loop or hardware in the loop
  - Generation of test cases and test vectors on model basis
  - Additional support by tools for static analysis of the source code
- Problem:
  - The model used for testing (verification and validation) shall be diverse to the model of the embedded system
    - If test cases and test vectors are based on the same model this will be the common cause for systematic failures, not detected by tests

Legal aspects (very short)

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- If manufacturer of tools for model-driven software engineering gives confirmation (insurance) about features related to functional safety, they will be liable too in case of dangerous failures of embedded systems, depending on faults of these features.
  - Because the user of the tool will trust these confirmation
- If a tool manufacturer does not give any confirmation regarding functional safety, he will be not liable for dangerous failures of embedded systems
  - The user of the tool has to qualify (verify and validate) the features of the tool regarding functional safety and will be liable by himself

Perspective

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- The tool chains for model-driven SWE will grow every year
  - Additional features regarding functional safety will be added
  - More and more will be formal specified and verified
  - Chip manufacturer will give support to tool manufacturer and will develop additional tools to close the gaps regarding functional safety or will implement safety integrity functions on chip in hardware to reduce software overhead
- Model-driven software engineering can be suitable for embedded systems regarding functional safety
  - to avoid systematic faults,
  - to get confidence about the correctness and completeness,
  - to shorten time to market

Perspective

- Together with TÜV SÜD, some chip manufacturer and tool manufacturer we are working on different concepts to come to solutions which can be accepted for embedded systems in the area of functional safety
- Goal should be a tool chain, which has the approval regarding functional safety, to come to efficient software engineering for embedded systems

Conclusion

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

- Further work regarding concepts to improve systematic safety integrity in verification and validation by model-driven testing (together with tool manufacturer)
- Working out concepts to improve safety integrity for embedded systems by model-driven software engineering depending on requirements of IEC 61508 and other functional safety standards
- Further work on hardware architectures of microcontroller to improve safety integrity on hardware level (together with chip industry)

- Thanks for your attention!

- Do you have further questions?

Contact

University of Passau
Institute of IT-Security and Security Law (ISL)
Innstr. 43
94032 Passau

Dipl.-Ing. Harald Hauff
Researcher

Tel.: +49 851 509-3026
Fax.: +49 851 509-3052
E-Mail: harald.hauff@uni-passau.de

Prof. Dr. Hermann de Meer
Chair of Computer Networks and Communication
+49 851 509-3051
+49 851 509-3052