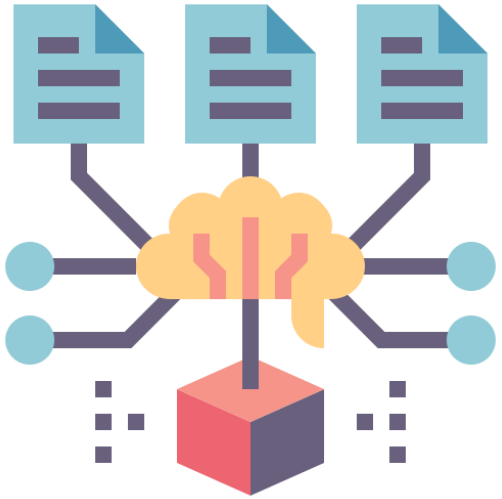
Systemengineering all inclusive

ein holistisches Konzept zur Entwicklung komplexer, sicherer eingebetteter Software



Thomas Barth

Prof. Dr. -Ing. Peter Fromm

REConf ®

h_da hochschule darmstadt

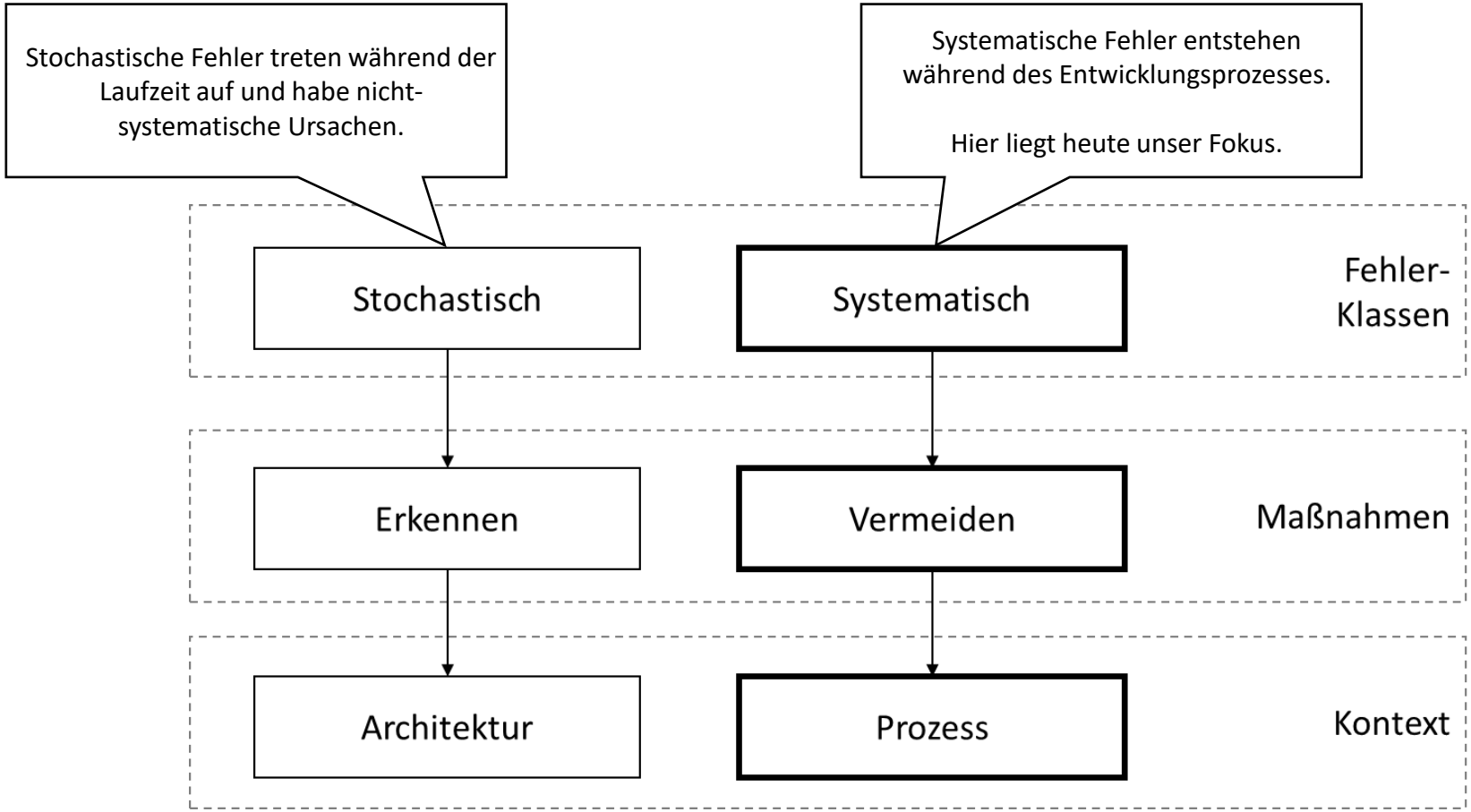
member of eut+

Agenda

- Kleiner Exkurs funktionale Sicherheit
- Warum ist es so schwer, die Spezifikation konsistent zu halten
- Motivation holistisches MBSE
- Vorstellung µRTE

Wo finden sich eingebettete Systeme die sicher funktionieren müssen ?

Wie entstehen Fehler welche die Sicherheit gefährden?



Wir müssen uns Gedanken machen was alles schief gehen kann

Hazard and Risk Analysis

erkannte Gefährdungen

Anforderungen

Wir müssen nachweisen, wie wir mit den erkannten Gefährdungen umgehen

Hazard and Risk Analysis

Safety case argument

erkannte Gefährdungen

Wurden die erkannten Risiken ausreichend reduziert?

Anforderungen

Safety case argument

Hazard and Risk Analyse

liefert

Top-level Requirements (Safety Goals)

ergeben

Technische Anforderungen

validieren

Tests

validieren

Hazard and Risk Analyse

Traceability
Die Kette muss nachvollziehbar und belegbar sein.

Zusätzliche Attribute
Womöglich müssen Attribute wie ein safety integrity level nicht nur dokumentiert sondern auch entlang der Kette weitergegeben werden.

Unterziehen sich

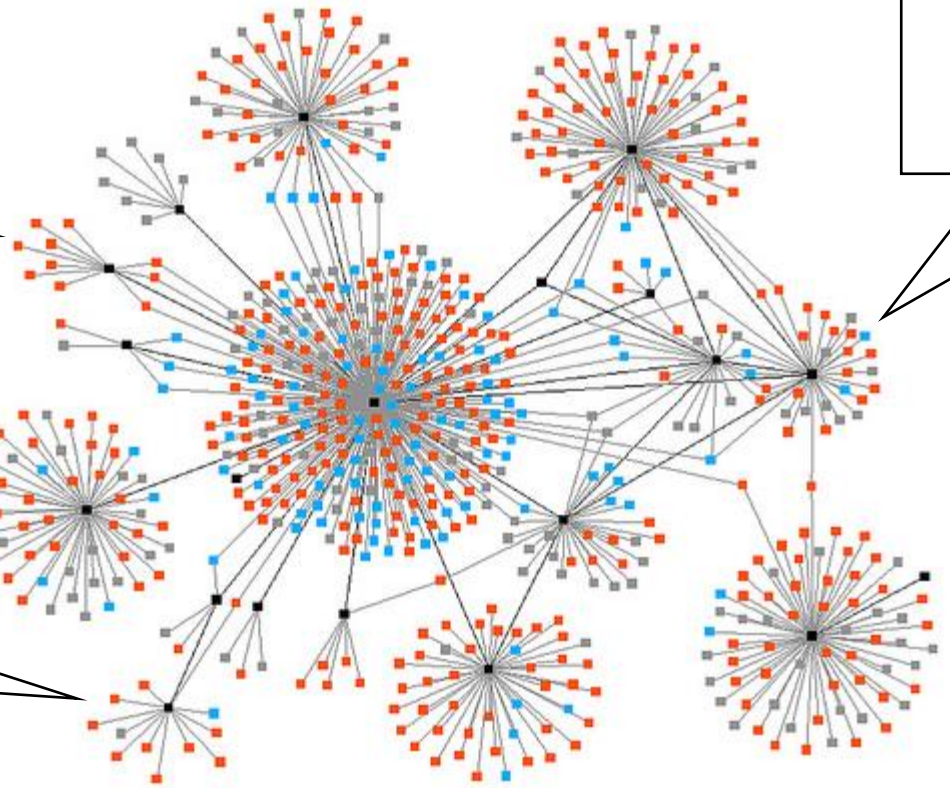
Implementierung

Dienen als Grundlage für

Problem1:
Komplexität ist die Summe von Elementen und deren Abhängigkeiten

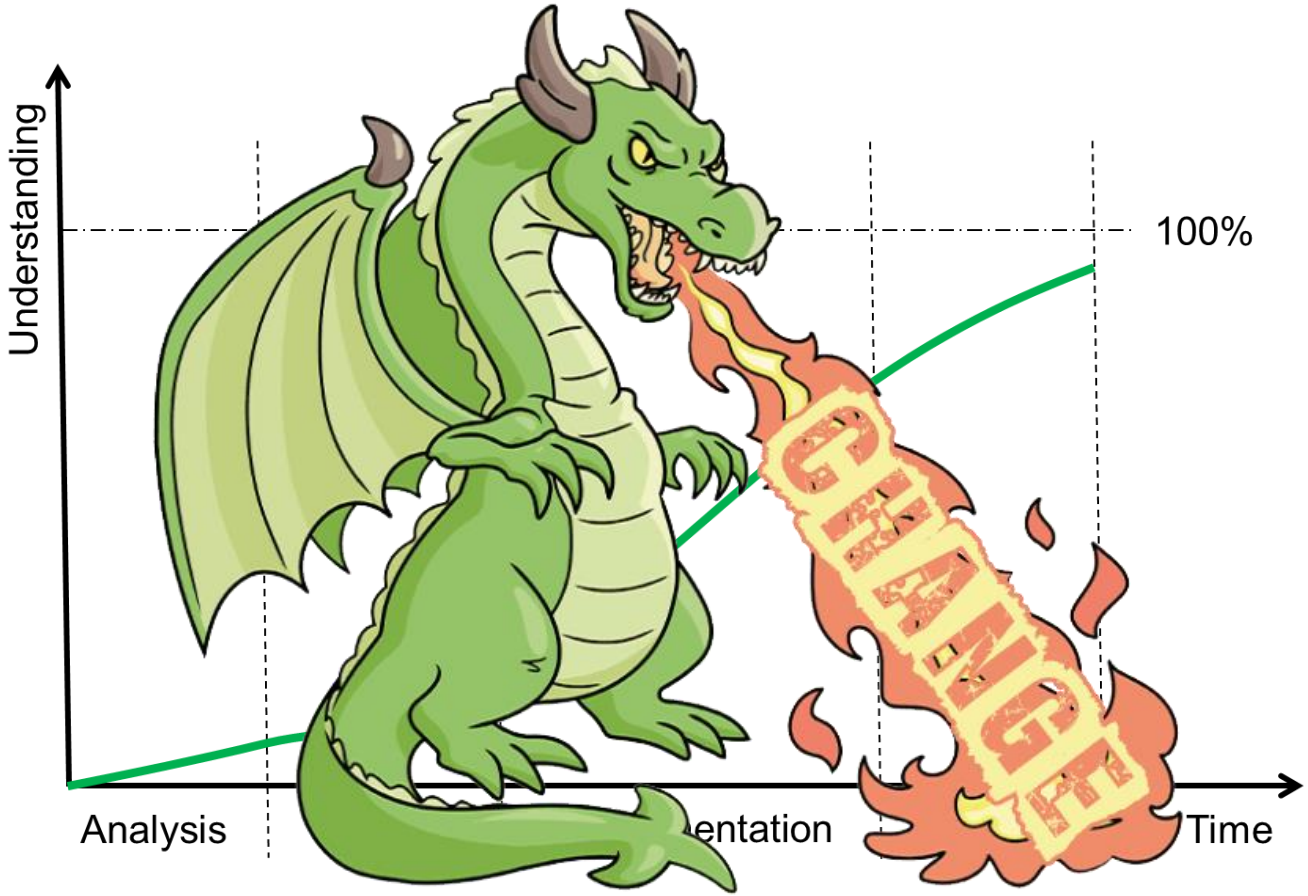
Kulturbrüche
Unterschiedliche Sichten auf das System, technisches Verständnis, Interessen.

Medienbrüche
Verschiedene Werkzeuge und Datenformate.



Organisationsbrüche
Unterschiedliche Teams, Abteilungen oder Organisationen.

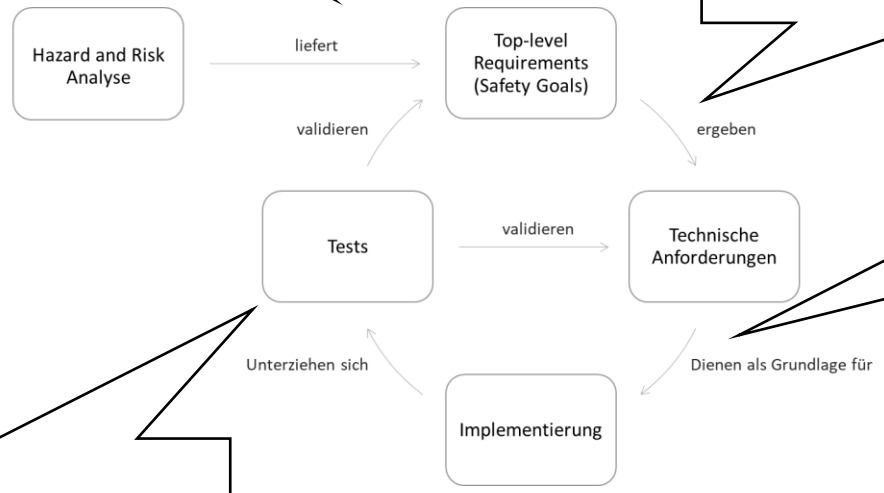
Problem 2:
Wir wissen erst am Ende des Projekts was wir hätten tun sollen



Typische Konsequenz – Verlust der Konsistenz

(Safety) Anforderungen
Wurden alle Anforderungen realisiert?

(Safety) Anforderungen
Wo wurden sie konkretisiert?



(Safety) Architektur
Wurden die Safety Funktionen auf der korrekten Hardware (CPU, Speicher) implementiert?

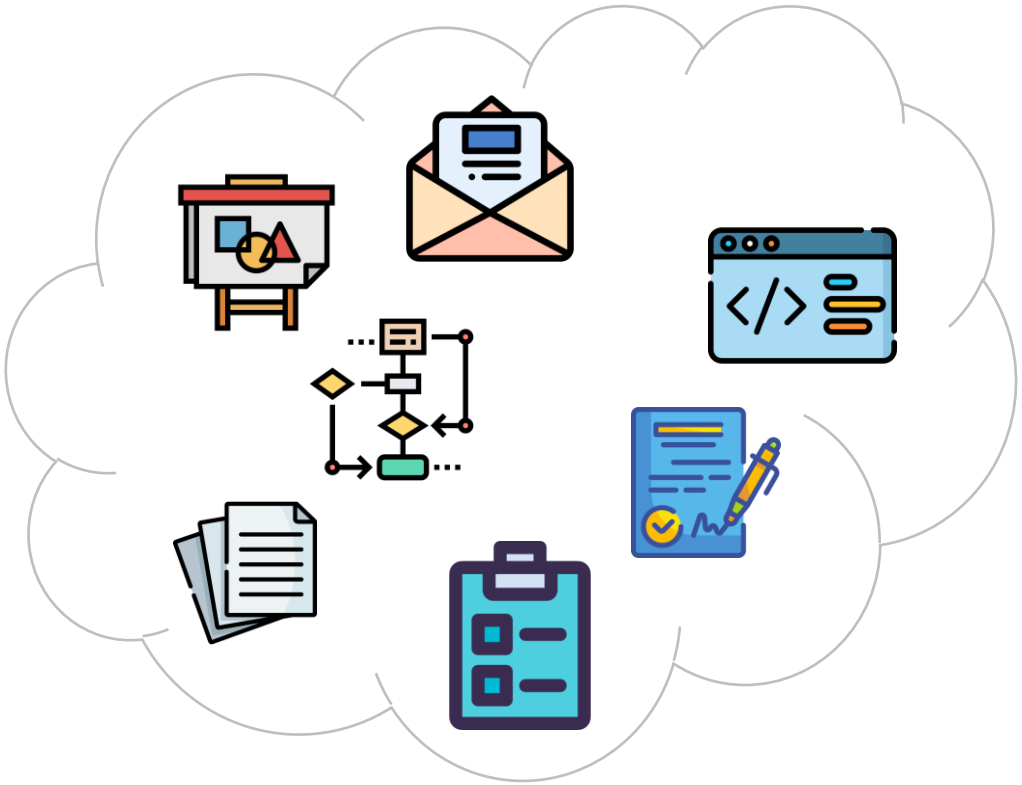
(Safety) Tests
Wurden alle Anforderungen getestet?

(Safety) Code
Wurde der Code ausreichend qualifiziert?

Wer weiß was MBSE ist?

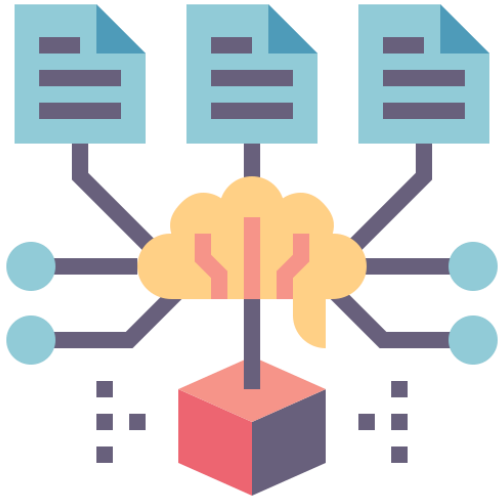
Wer wendet MBSE an?

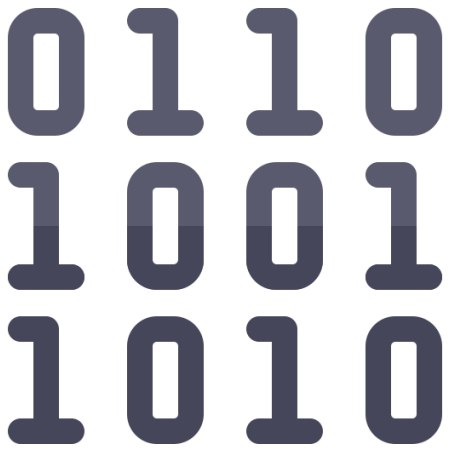
Unser Ansatz: SE all inklusive in einem Modell



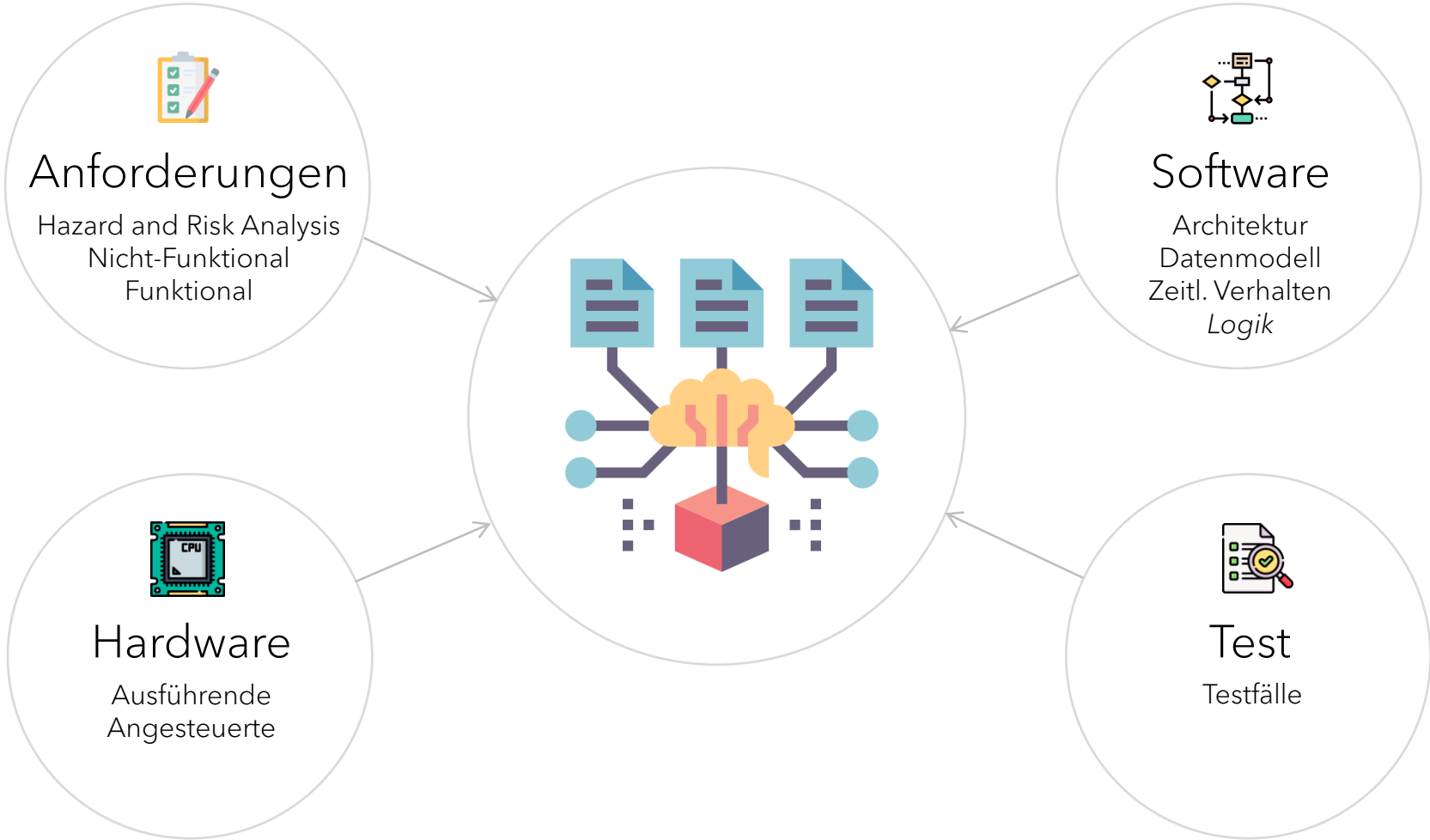
0 1 1 0
1 0 0 1
1 0 1 0

Unser Ansatz: SE all inklusive in einem Modell

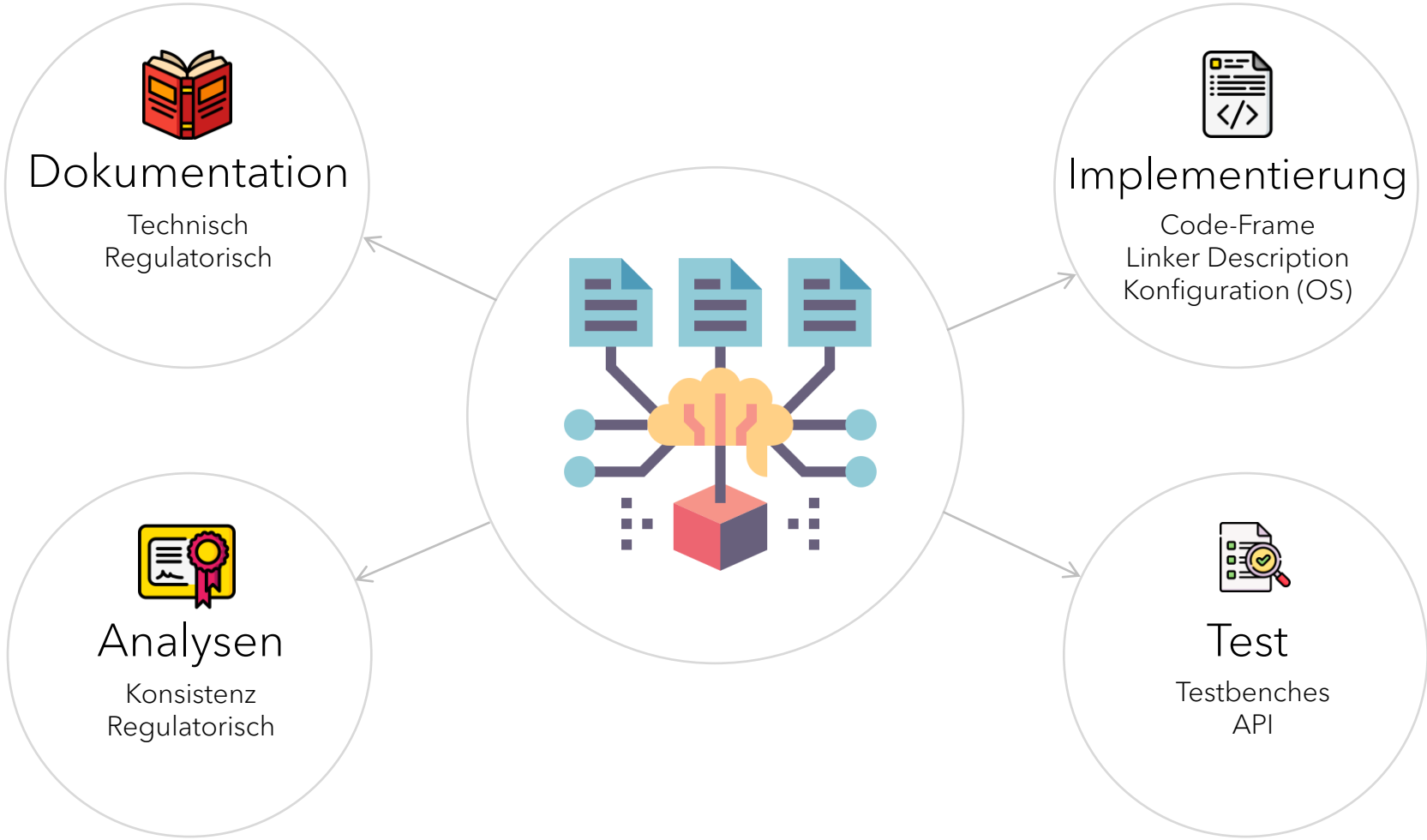




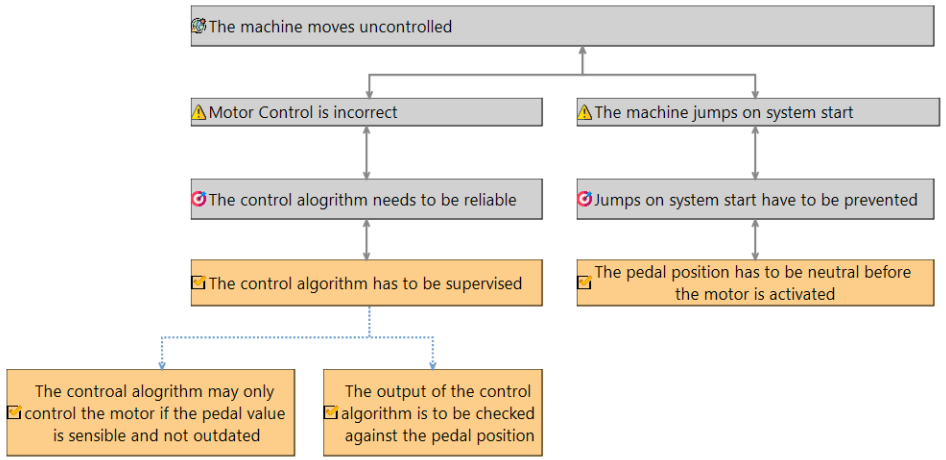
Welche Systemaspekte werden benötigt?



Was kann aus dem Modell abgeleitet werden?



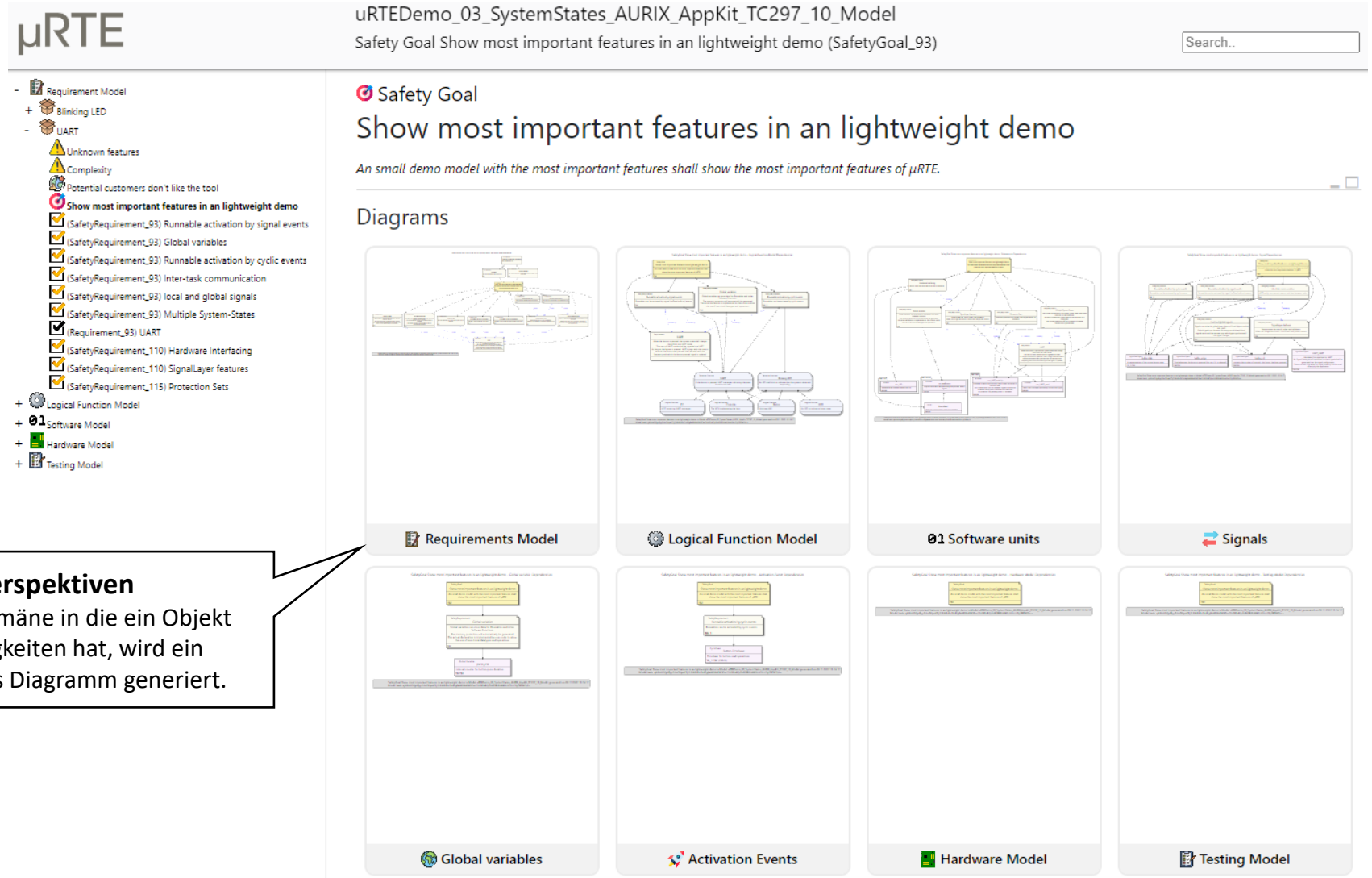
µRTE Modellierung von Anforderungen



- µRTE System Model
  - Requirement Model
    - Package Main
      - Hazardous Event Motor Control is incorrect
      - Hazardous Event The machine jumps on system start
      - Hazard Scenario The machine moves uncontrolled
      - Safety Goal The control alogrithm needs to be reliable
      - Safety Goal Jumps on system start have to be prevented
      - Safety Requirement (SafetyRequirement_149) - The control algorithm has to be supervised
        - refined by The controal alogrithm may only control the motor if the pedal value is sensible and not outdated
        - refined by The output of the control algorithm is to be checked against the pedal position
      - Safety Requirement (SafetyRequirement_150) - The pedal position has to be neutral before the motor is activated
      - Safety Requirement (SafetyRequirement_151) - The output of the control algorithm is to be checked against the pedal position
      - Safety Requirement (SafetyRequirement_152) - The controal alogrithm may only control the motor if the pedal value is sensible and not outdated
  - Logical Function Model
  - Software Model
  - Hardware Model
  - Testing Model

	User-ID	Name	Description	Status	SIL	SIL derived	SIL manual	SIL effective
Package Main	ReqPackage_143	Main						
Hazard Scenario The machine moves uncontrolled	HazardScenario_144	The machine moves uncontrolled	The machine accumultes enough kintic energy to endanger its enviror					
Hazardous Event Motor Control is incorrect	HazardousEvent_145	Motor Control is incorrect			SIL_4			
Hazardous Event The machine jumps on system start	HazardousEvent_146	The machine jumps on system start	The pedal is not in neutral upon system start and the machine jumps		SIL_3			
Safety Goal Jumps on system start have to be prevented	SafetyGoal_148	Jumps on system start have to be prevented				SIL_3	derived	SIL_3
Safety Goal The control alogrithm needs to be reliable	SafetyGoal_147	The control alogrithm needs to be reliable				SIL_4	derived	SIL_4
uirement (SafetyRequirement_149) - The control algorithm has to be	SafetyRequirement_149	The control algorithm has to be supervised		implemented		SIL_4	derived	SIL_4
afetyRequirement_150) - The pedal position has to be neutral before	SafetyRequirement_150	The pedal position has to be neutral before the motor is a		implemented		SIL_3	derived	SIL_3
equirement_151) - The output of the control algorithm is to be checke	SafetyRequirement_151	The output of the control algorithm is to be checked agair		implemented		SIL_4	derived	SIL_4
nt_152) - The controal alogrithm may only control the motor if the pe	SafetyRequirement_152	The controal alogrithm may only control the motor if the p		implemented		SIL_4	derived	SIL_4

µRTE Dokumentation – Übersicht Doku



Perspektiven
Für jede Domäne in die ein Objekt Abhängigkeiten hat, wird ein spezifisches Diagramm generiert.

µRTE Dokumentation – Tabellarische Sicht

Software Layer

Software units (5)

Sofware units associated with Safety Requirements this Safety Goal links to.

Hidden columns:
» Function calls » Technical Functions » Type » Tasks » WCET » Stack » ROM » Globals » ProtectionSets » SIL req » SIL ach » sub Technical Functions (R) » sub Requirements (R) » Has a return value (R) » SystemStates (R) » Ingoing Trigger Ports (R) » Outgoing Trigger Ports (R) » Ingoing Data Ports (R) » Outgoing Data Ports (R) » Runnables (D) » Is Synchronous (D) » Hardware (D) » Ports (G) » Callers (F) » Return Type (F) » Parameters (F)

Unit	Parent	Requirements	Signals (D)	DataType (D)
ButtonRead Reads the current button state from hardware	*Button*	*(SafetyRequirement_110) Hardware Interfacing* *(SafetyRequirement_110) SignalLayer features* *(Requirement_93) UART*	*HWI_Button*	T *uRTE_boolean_t*
drv_LED Hardware write-Interface towards the LED	*LED*	*(SafetyRequirement_110) Hardware Interfacing* *(SafetyRequirement_110) SignalLayer features* *(Requirement_93) UART*	*HWI_LED*	T *uRTE_boolean_t*
run_UART_send Sends UART messages periodically via the UART signal	*UART*	*(Requirement_93) UART* *(SafetyRequirement_110) SignalLayer features* *(Requirement_93) UART* *(SafetyRequirement_93) Multiple System-States* *(Requirement_93) UART*	-	-
run_UART_wakeUp Runnable to switch into the UART state if there is an event in the Blink State. This runnable does not use hardware signals but accesses hardware directly and is therefore associated with an protection set grandting access to hardware.	*UART*	*(Requirement_93) UART* *(SafetyRequirement_115) Protection Sets* *(Requirement_93) UART* *(SafetyRequirement_93) Multiple System-States* *(Requirement_93) UART*	-	-
run_readButton Acquires the button state periodically and provides button signals	*Button*	*(SafetyRequirement_93) Global variables* *(Requirement_93) UART*	-	-

Signals (3)

Signals of the (Safety)Requirements.

Global Variables (1)

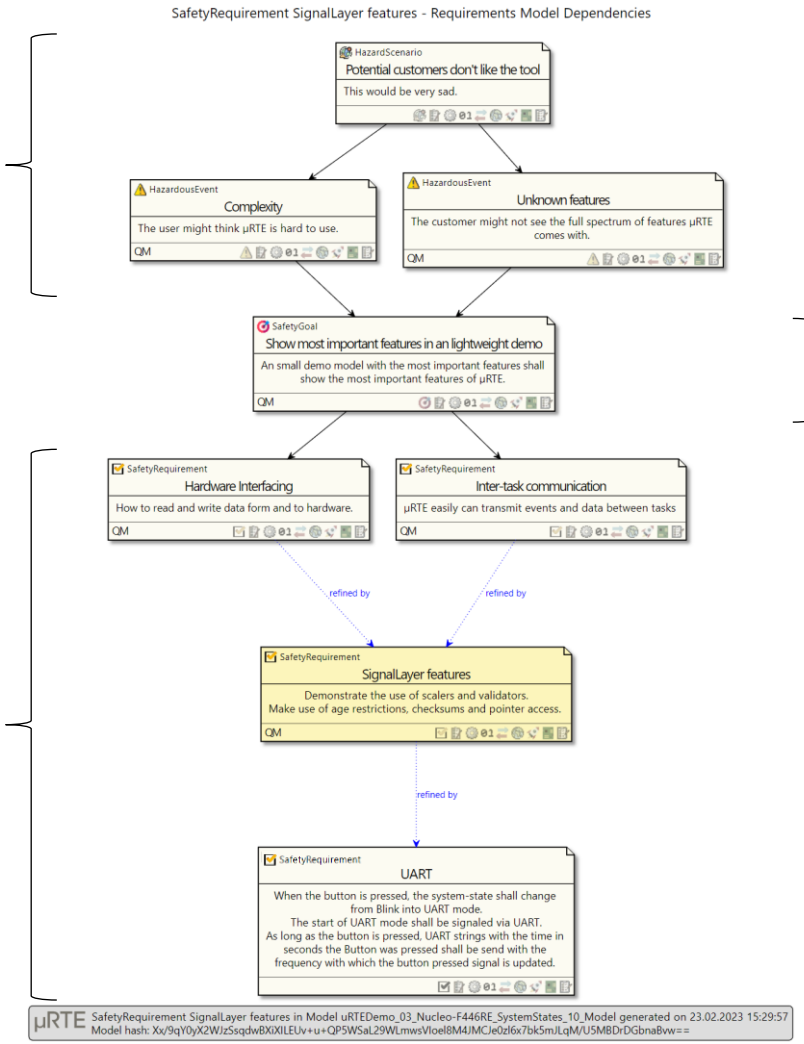
Global variables of the (Safety)Requirements.

Activation-Events (1)

Activation Events of the (Safety)Requirements.

µRTE Dokumentation – Beziehung zu Anforderungen

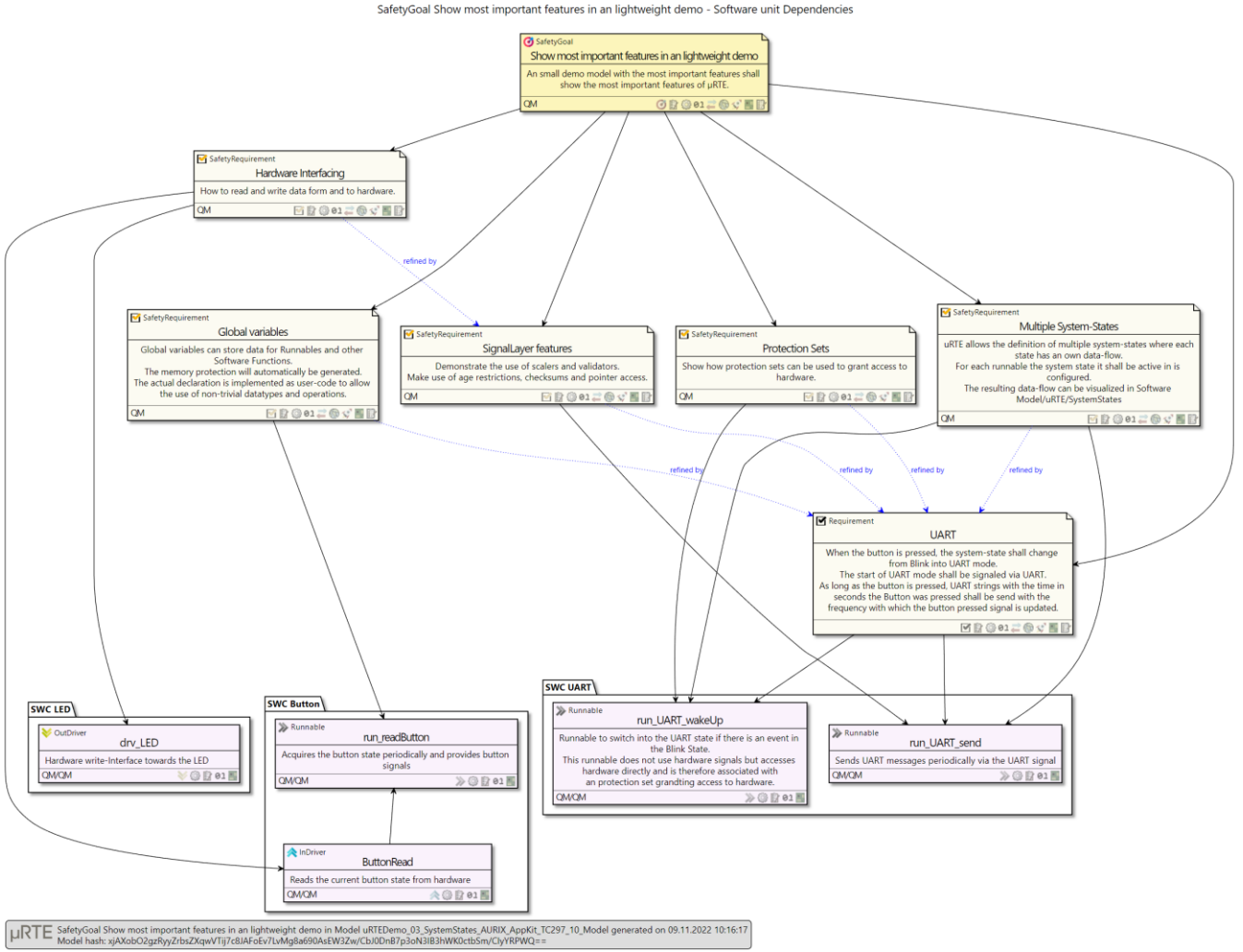
Hazards and Risks
Hazard scenario und
Hazardous events



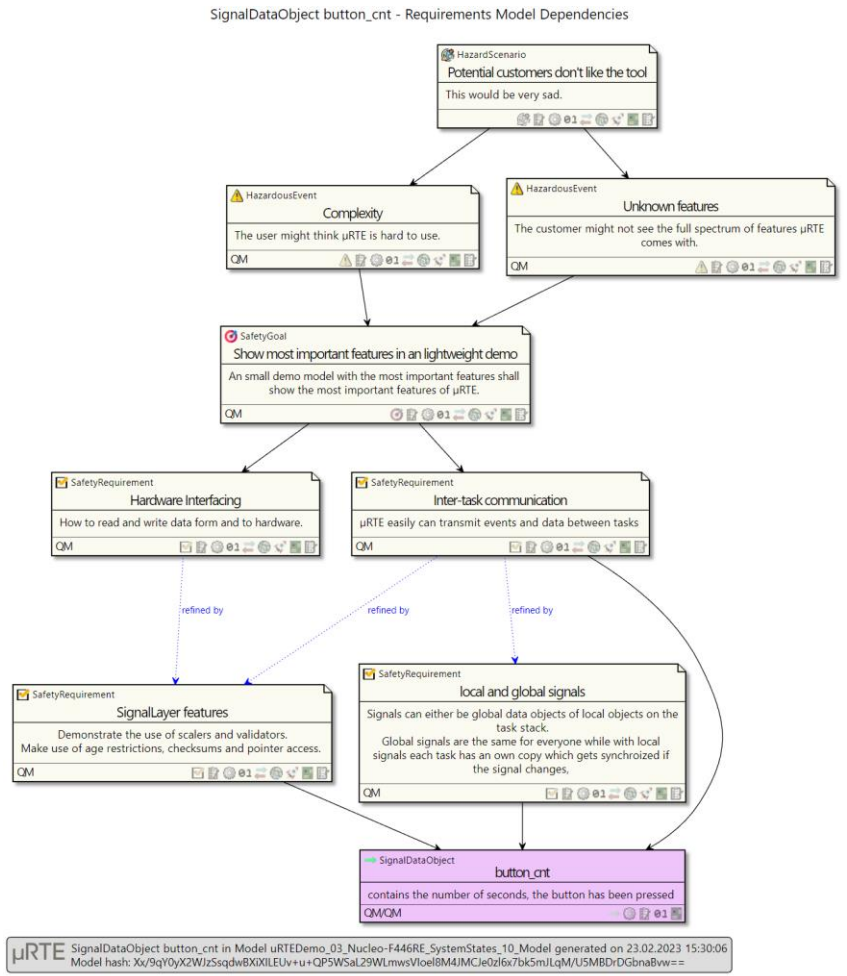
Top-level Anforderung
Auch als Safety-Goal bezeichnet

Anforderungern
Anforderungen konkretisieren sich
bis zu technischen Anforderungen

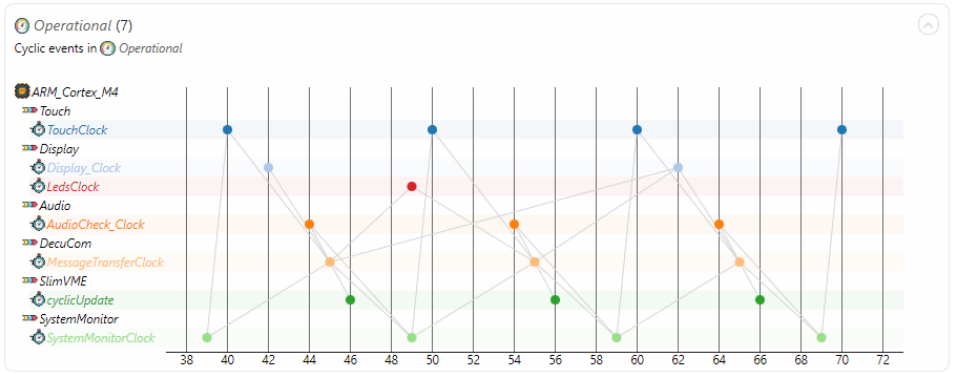
Argumentationskette
Anforderung zu Implementierung

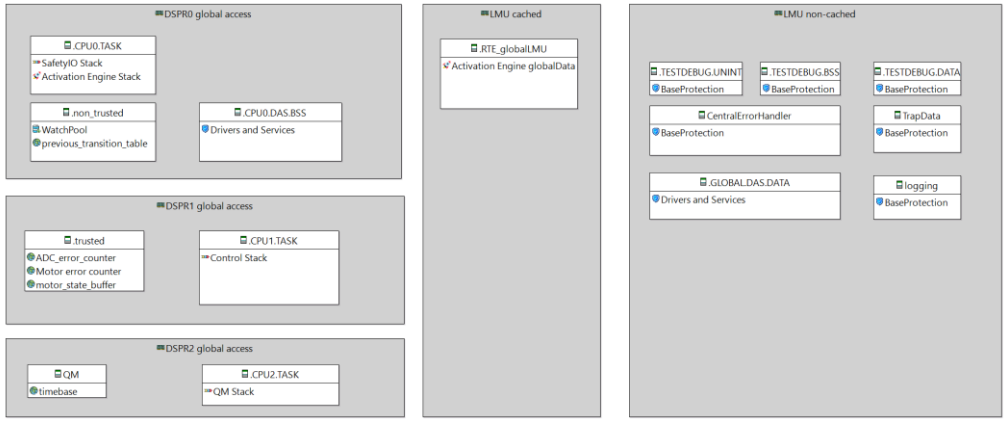


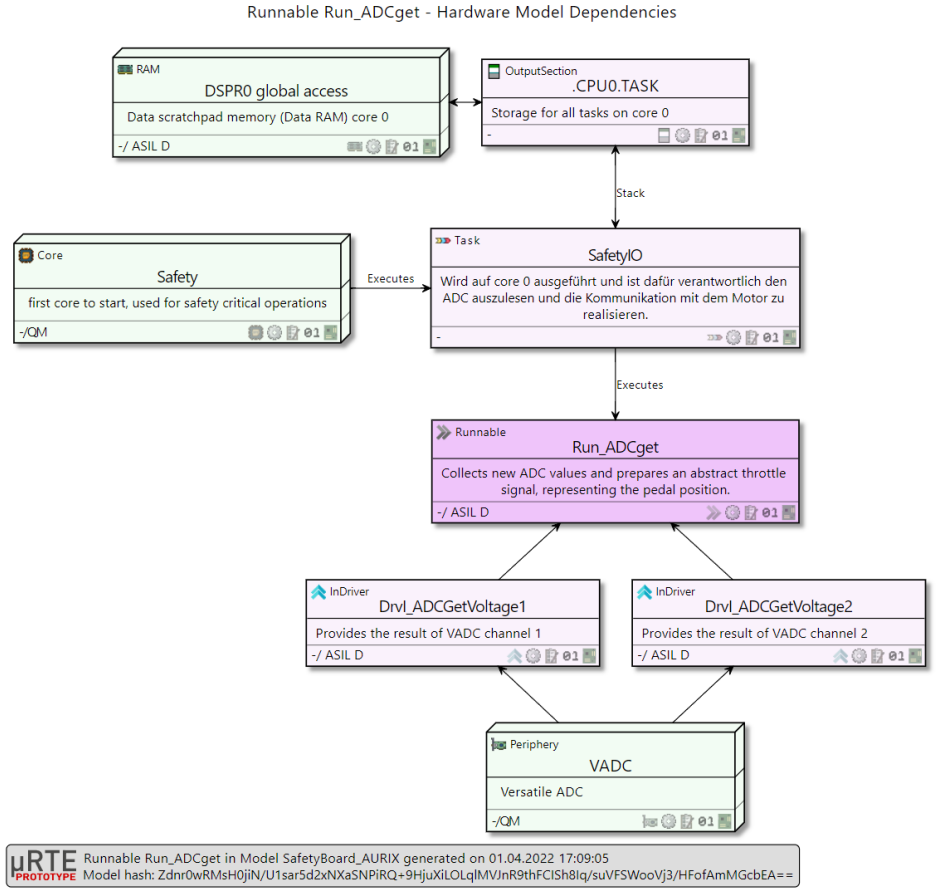
Argumentationskette
Implementierung zu Anforderungen



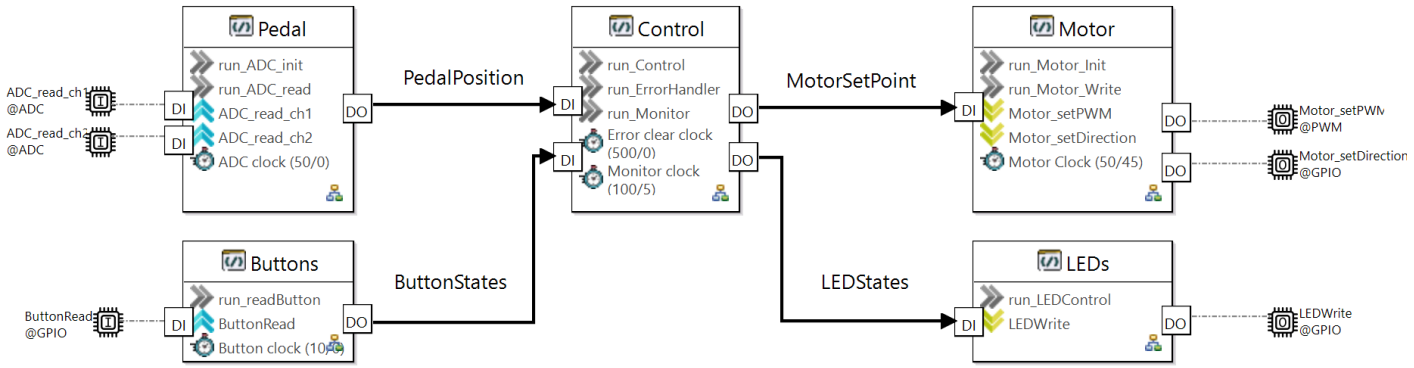
Weitere µRTE Darstellungen



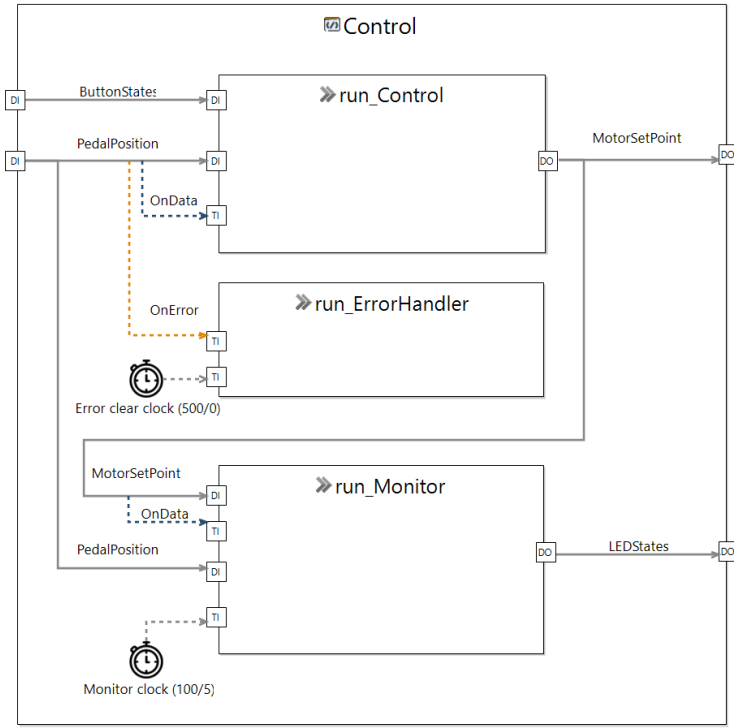




µRTE Software Modellierung



What you see is what you get
Keine Brüche da Code direkt aus dem Modell erzeugt wird.



µRTE Code

Kontext
Wo sind wird im System und wann wird der Code aufgerufen

Anforderungen
Was soll der Code an dieser Stelle machen und was ist zu beachten.

Spezifische Signatur
Alle Datenquellen und Senken werden durch Signale abstrahiert. Der Code ist unabhängig vom Restsystem.

```
/* ------------ Runnable "R    CErrorHandler" ------------
 * This error handler handles A  errors of the connected throttle paddle.
 * It decides whether the system shall go into an safe state or remain operational.
 *
 *  User-ID:        Runnable_18
 *  SystemStates:   "operational"
 *  WCET:           10µs
 *  Stack:          250 Byte
 *  ROM:            110 Byte
 *
 * --- Triggers ---
 *      - TriggerPort "ErrorCounterClock"
 *          Task: "SafetyIO"
 *          System Events:
 *              - Cyclic Trigger "Error Counter clock" Cycle Time: 300, Offset 0
 *
 *      - TriggerPort "RunTPort_ADCErrorHandler"
 *          Task: "SafetyIO"
 *          Signals:
 *              - "ADC1" Trigger-Type: onError
 *              - "ADC2" Trigger-Type: onError
```

```
* --- SAFETY ---
*      SIL manual:     derived
*      SIL effective:   SIL_1
*      SIL achieved:   QM
*
* --- Requirements ---
*          - "Translate targetspeed into engine speed"
*              Description:
*                  The targetspeed, which comes as a 3 value vector vx, vy and vphi must be translated into 4 individual engine speeds.
*
* --- SafetyRequirements ---
*          - "Explicit driving state" SIL effective: derived
*              Description:
*                  The engines may only be active in the driving state. In all other states, the engines must stop.
*              Other References of the SafetyRequirement:
*                  - Runnable Monitor_cyclicCheck_run
*          - "Limit speed" SIL effective:  SIL_1
*              Description:
*                  The resulting engine speed may never be faster that the allowed speed limit determined by the monitor functions.
*          - "Slow down / Stop car" SIL effective:  SIL_1
*              Description:
*                  The car shall slow or stop down if the obstacle is far away. The distance depends on the current speed.
*              Other References of the SafetyRequirement:
*                  - Runnable Monitor_cyclicCheck_run
*
```

```
void ENGINE_setSpeed_run(

        /* Trigger */
        const uRTE_Triggers_t triggerPort,

        /* Incoming Data-Signals */
        Signal_CONTROL_limits& sig_IN_CONTROL_limits,   /* Structure containing driving limiting parameters like maxSpeed */
        Signal_CONTROL_targetspeed& sig_IN_CONTROL_targetspeed, /* The car targetspeed */

        /* Outgoing Data-Signals */
        Signal_FAULHABER_speed_out* const p_sig_OUT_FAULHABER_speed_out /* Interface towards the speed object on the motion controller */
){

    //local IN signal value buffers
    Signal_CONTROL_limits::data_t CONTROL_limits_data;
    Signal_CONTROL_targetspeed::data_t CONTROL_targetspeed_data;

//Start of user code implementation ENGINE_setSpeed_run
```

µRTE Analyse

Safety-Requirement - SafetyRequirement_110

Hardware Interfacing

How to read and write data form and to hardware.

Warnings

Testing Warnings (1)

Testing Warnings for SafetyRequirement Hardware Interfacing
Testing warnings are related to the tests in the testing layer and their depedencies.

(SafetyRequirement_110) Hardware Interfacing is not referencing a test and not all refinements reference a test.

Requirements Warnings (1)

Requirements Warnings for SafetyRequirement Hardware Interfacing
Requirements warnings are related to the requirements layer.

(SafetyRequirement_110) Hardware Interfacing is not rejected or implemented.

Beispiel Reports
Auf u-RTE.com oder heute und morgen persönlich

Problems | Console

0 errors, 76 warnings, 0 others

Description	Location
Warnings (76 items)	
Core Arm Cortex®-M7 (QM) was selected for execution for ActivationEngine which does not	ActivationEngine
CyclicEvent CAN_dispatch_clock is not associated with any trigger port.	CyclicEvent_CAN_dispatch_clock
DataType bool is unused.	DataType_bool
DataType CAN_RX_MSG_t is unused.	Type_CAN_RX_MSG_t
DataType int16_t is unused.	e_int16_t
DataType sint32_t is unused.	sint32_t
DataType uint32_t is unused.	t32_t
ECU PSOC SmartPower has a SIL effective of SIL_1 but a SIL achieved of QM	rtPower
Function DRIVE_0_calibration_fun is not directly associated with a Technical Function	_calibration_fun

ieved of QM

of QM

ART

LED, UART

king LED, UART

xed SILs in .button : QM, SIL_1.

.button needs a SIL of SIL_1 but is using main for storage which has a achieved SIL of QM.

.rtos.task.Button needs a SIL of SIL_1 but is using main for storage which has a achieved SIL of QM.

.rtos.task.LED needs a SIL of SIL_1 but is using main for storage which has a achieved SIL of QM.

.rtos.task.UART needs a SIL of SIL_1 but is using main for storage which has a achieved SIL of QM.

.rtos.task.uRTE needs a SIL of SIL_1 but is using main for storage which has a achieved SIL of QM.

.uRTE needs a SIL of SIL_1 but is using main for storage which has a achieved SIL of QM.

Memory referenced by main for the stack of Button is using main with a SIL achieved of QM, which does not satisfy the SIL required of SIL_1 for this task.

Memory referenced by main for the stack of LED is using main with a SIL achieved of QM, which does not satisfy the SIL required of SIL_1 for this task.

Memory referenced by main for the stack of UART is using main with a SIL achieved of QM, which does not satisfy the SIL required of SIL_1 for this task.

Memory referenced by main for the stack of ActivationEngine is using main with a SIL achieved of QM, which does not satisfy the SIL required of SIL_1 for the central activation engine.

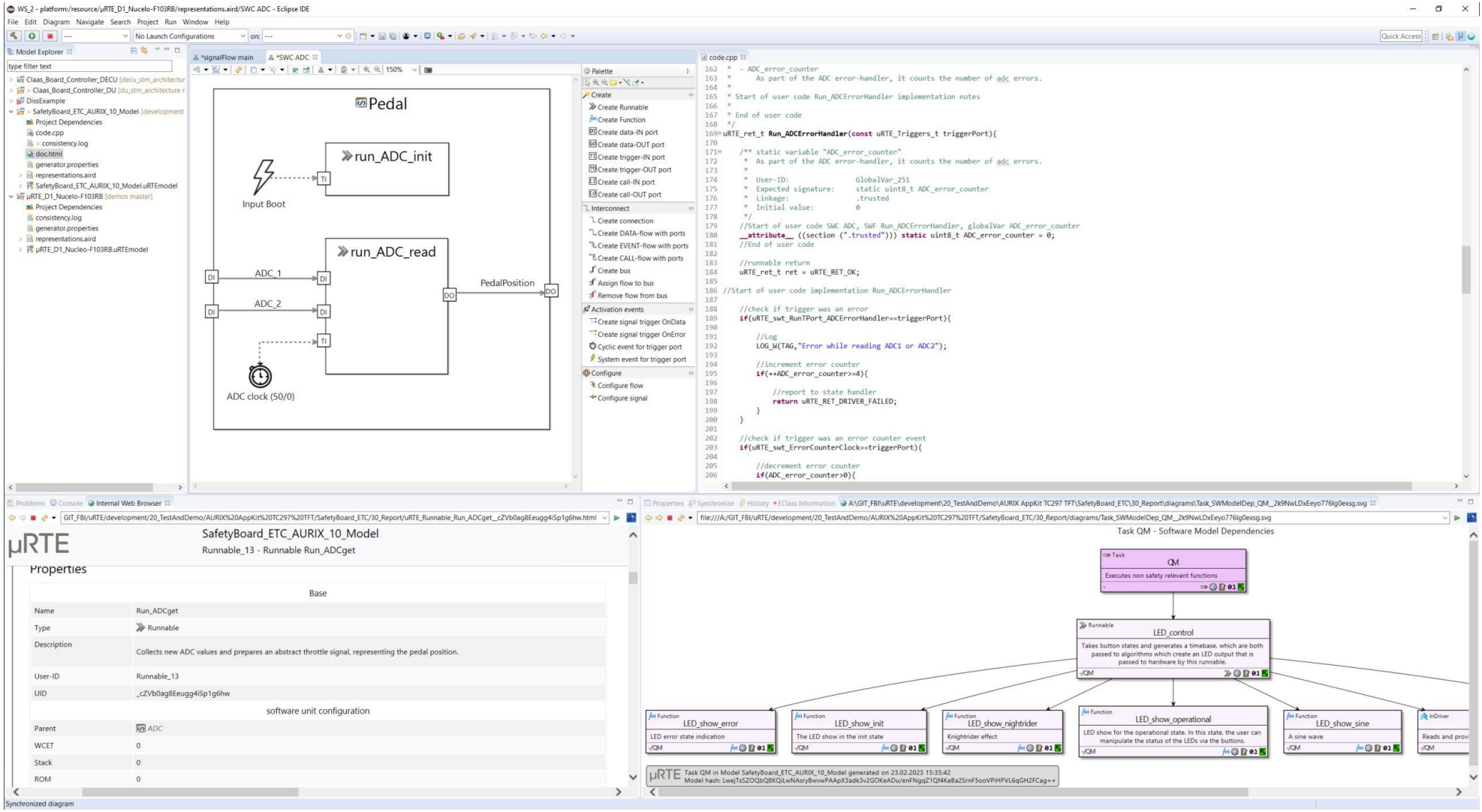
Button needs a SIL of SIL_1 but is executing on Arm® Cortex®-M4 which has a achieved SIL of QM.

LED needs a SIL of SIL_1 but is executing on Arm® Cortex®-M4 which has a achieved SIL of QM.

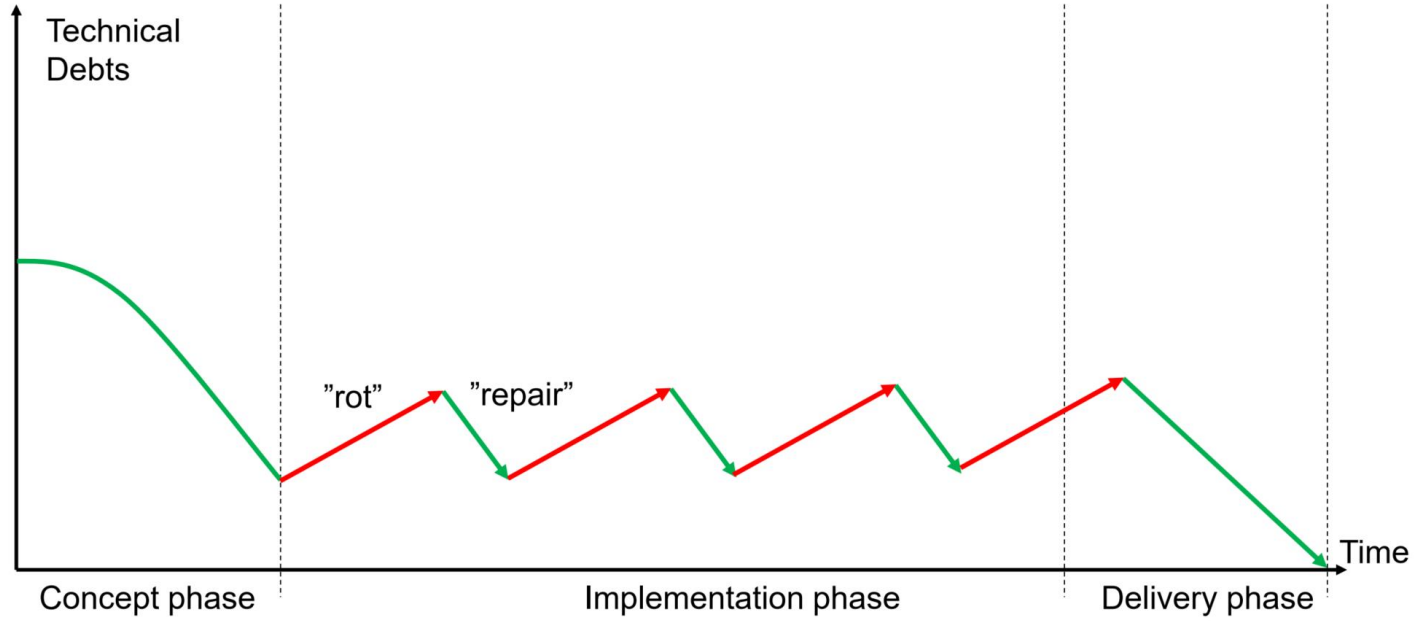
UART needs a SIL of SIL_1 but is executing on Arm® Cortex®-M4 which has a achieved SIL of QM.

Arm® Cortex®-M4 (QM) was selected for execution for ActivationEngine which does not satisfy the SIL required of SIL_1.

Touch and feel µRTE



Inkrementeller/Iterativer Prozess über ALLE Artefakte



Zusammenfassung

Was wir erreicht haben

- Holistisches Model
- Iteratives Arbeiten wird unterstützt
- Ausführbare Laufzeitumgebung kann generiert werden
- Hilfreiche Analysen
- Ausführlicher Report
- Derzeit Safety Assessment in industriellem Projekt nach ISO 25119

Limitierungen

- Fokus auf Signalfluss getriebene Systeme
- Ursprünglich ein Forschungsprojekt welches nun aber auch industriell genutzt wird
- Kollaboration nur bedingt durch das Framework unterstützt
- Usability leidet bei umfangreichen Projekten

Ausblick

- Industrielle Nutzung soll ausgebaut werden
- Möglichkeiten der verbesserten Kollaboration sollen erforscht werden
- Erweiterte Analysemöglichkeiten, z.B. Vergleich gegen map-file, statische Codeanalyse oder trace

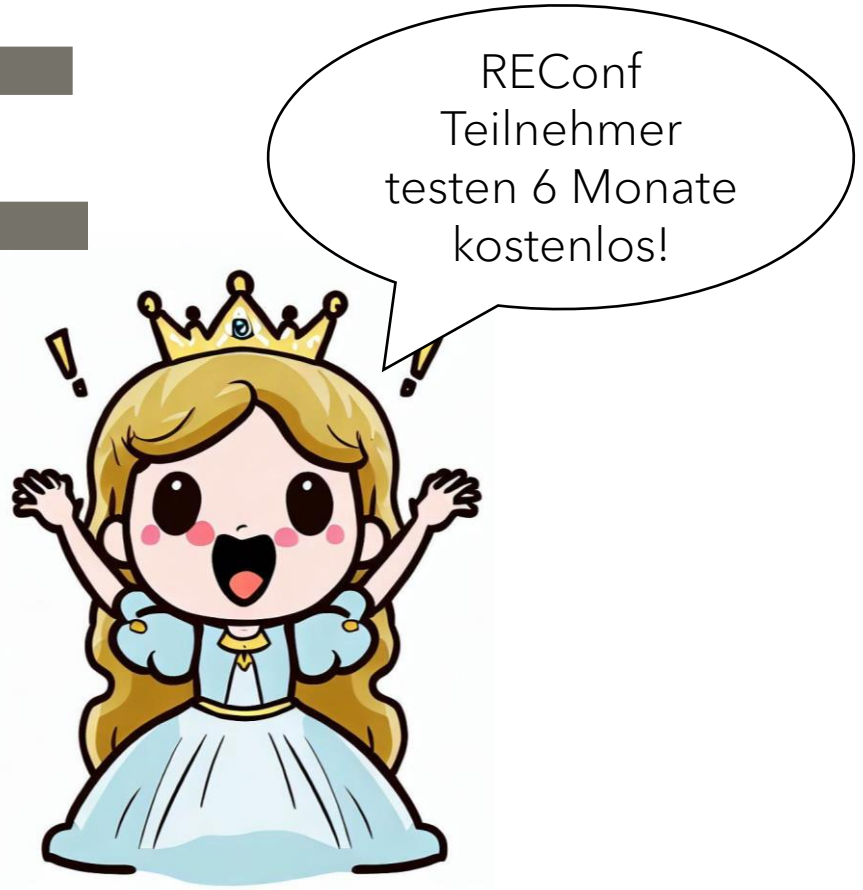
Fragen

Thomas Barth
✉ thomas.barth@h-da.de

Prof. Dr. -Ing. Peter Fromm
✉ peter.fromm@h-da.de

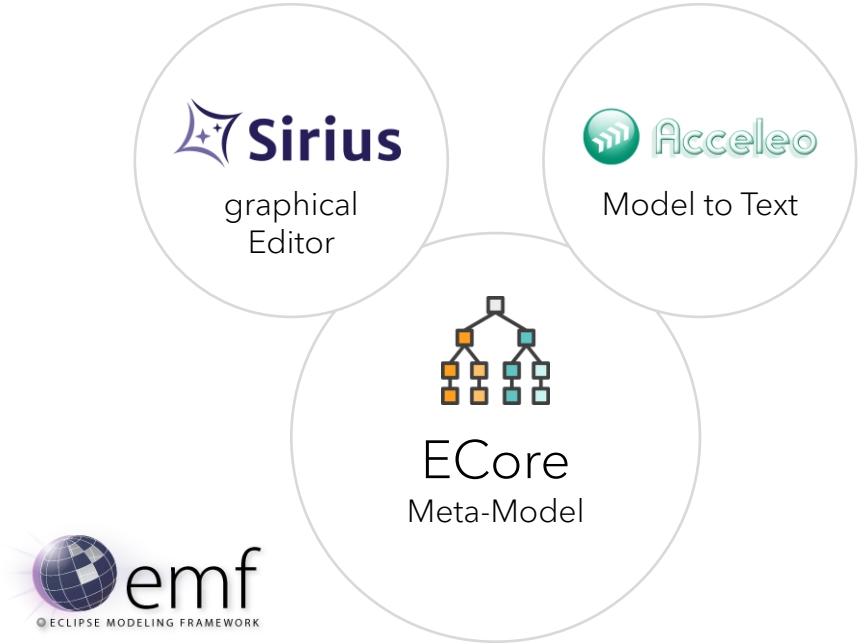
µRTE

u-RTE.com

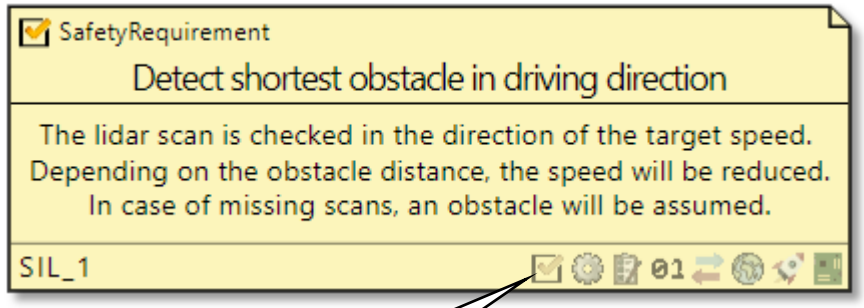


Backup

Eclipse Modeling Framework

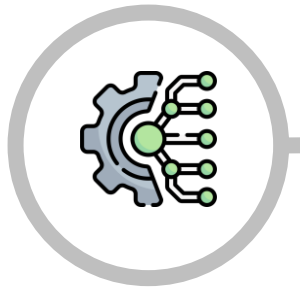


µRTE Dokumentation - Elemente



Grafische Navigation
Jedes Element hat einen eigenen Report, die jeweiligen Diagramme können direkt angesprungen werden

Der µRTE Workflow



Integration

Die Abstraktionsschicht für µRTE ist für jedes Target und jedes RTOS einmalig zu definieren

Anforderungen

Das gewünschte Verhalten des Systems wird definiert
Sicherheitsanforderungen werden aus den anzuwendenden Normen abgeleitet

Iteratives (agiles) Arbeiten wird unterstützt
Sogar späte Änderungen an der Architektur sind möglich

Architektur

Hardware, Datenflüssen und generelle Strukturen werden festgelegt

Generation

Dokumentation und Code/Laufzeitumgeben werden generiert

Implementierung

Die Logik wird innerhalb der generierten Laufzeitumgebung unter Verwendung der Dokumentation implementiert

Test and Qualifikation

Unit-, Integrations- und System-Tests werden auf Basis der Dokumentation und unterstützt durch uniforme Schnittstellen durchgeführt

Herausforderung HW/SW Interface