Architekturvarianten sicherheitskritischer Echtzeitsysteme

Sichere und kostengünstige Lösungen systematisch ermitteln

Ulrich Becker
Method Park

Isabella Stilkerich
Schaeffler Technologies

Ralf Münzenberger
INCHRON

Referenten

Dr. Ulrich Becker

Trainer, Berater und Coach bei Method Park Consulting GmbH
Software-Architektur, ALM, Verbesserung von Entwicklungsprozessen
https://xing.to/ubr

Dr. Isabella Stilkerich

Software-Architektin und Forscherin im Bereich E-Mobilität
Echtzeitsysteme, eingebettete Systeme, funktionale Sicherheit und AUTOSAR. Forscherin und SW-Architektin im BMBF Projekt ARAMiS II.

Dr.-Ing. Ralf Münzenberger

Verantwortlich für den Bereich Professional Services. In mehr als 160 Projekten hat er Kunden rund um das Thema Timing und Performance, Architekturoptimierung und funktionale Sicherheit beraten.

Weitere Autoren

Der Vortrag basiert auf dem Fachartikel „**Kostengünstig? Aber sicher!**“, Hanser Automotive 10/2017

Weitere Autoren des Fachartikels:

Christian Lederer

Teamleiter

Method Park Engineering GmbH

Frank Pinecker

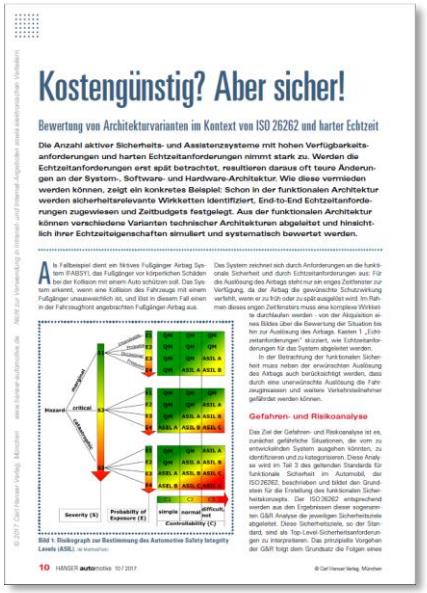
Berater

Method Park Consulting GmbH

Philip Rehkop

Professional Services Engineer

INCHRON GmbH



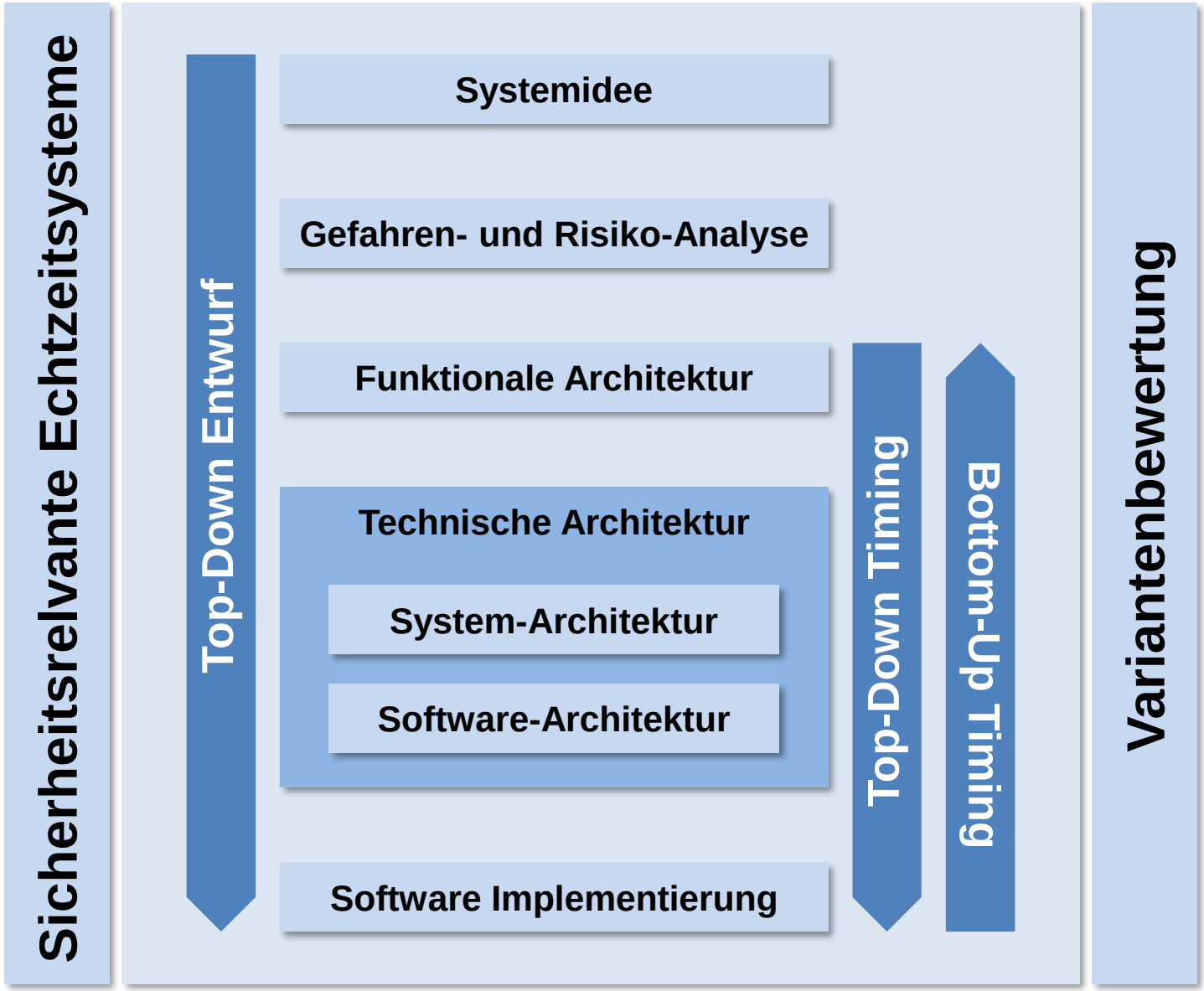
Kostengünstig? Aber sicher!

Bewertung von Architekturvarianten im Kontext von ISO 26262 und harter Echtzeit

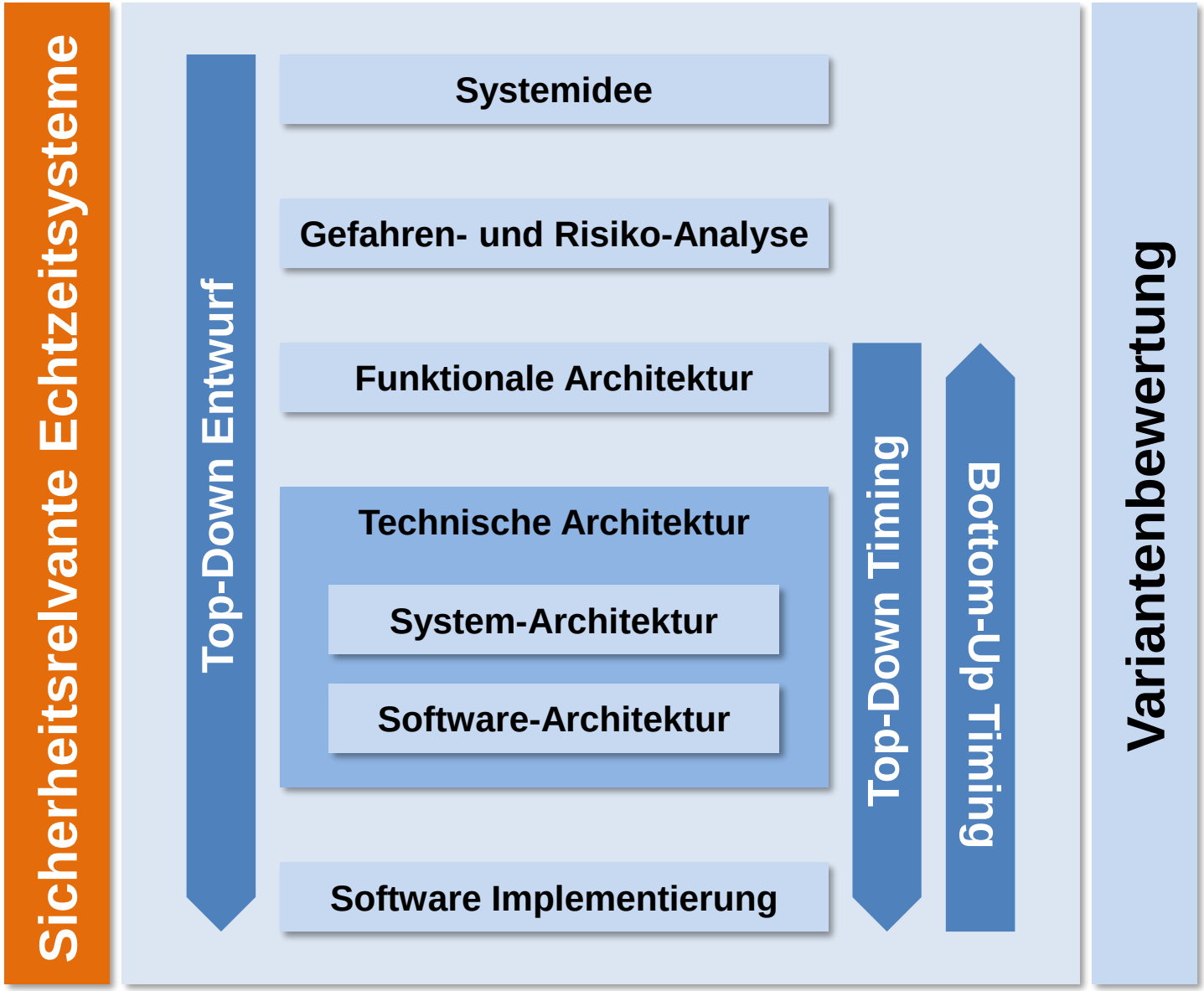
Die Anzahl aktiver Sicherheits- und Assistenzsysteme mit hohen Verfügbarkeitsanforderungen und harten Echtzeitanforderungen nimmt stark zu. Werden die Echtzeitanforderungen erst spät betrachtet, resultieren daraus oft teure Änderungen an der System-, Software- und Hardware-Architektur. Wie diese vermieden werden können, zeigt ein konkretes Beispiel: Schon in der funktionalen Architektur werden sicherheitsrelevante Wirkketten identifiziert, End-to-End Echtzeitanforderungen zugewiesen und Zeitbudgets festgelegt. Aus der funktionalen Architektur können verschiedene Varianten technischer Architekturen abgeleitet und hinsichtlich ihrer Echtzeiteigenschaften simuliert und systematisch bewertet werden.

Gefahren- und Risikoanalyse

Überblick


Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

Überblick


Sicherheitsrelevante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

Echtzeitsystem

DIN 44300

Echtzeitbetrieb ist ein Betrieb eines Rechensystems, bei dem Programme zur Verarbeitung anfallender Daten ständig betriebsbereit sind derart, dass die Verarbeitungsergebnisse innerhalb einer vorgegebenen Zeitspanne verfügbar sind.

Allgemein:

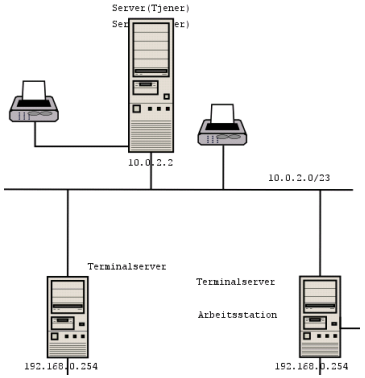
Echtzeitbetrieb bedeutet Rechtzeitigkeit.

Korrektes Systemverhalten hängt nicht nur von den Berechnungsergebnissen ab, sondern auch vom Zeitpunkt der Erzeugung und Verwendung dieser Ergebnisse.

Echtzeitsystemklassen

Weiches Echtzeitsystem

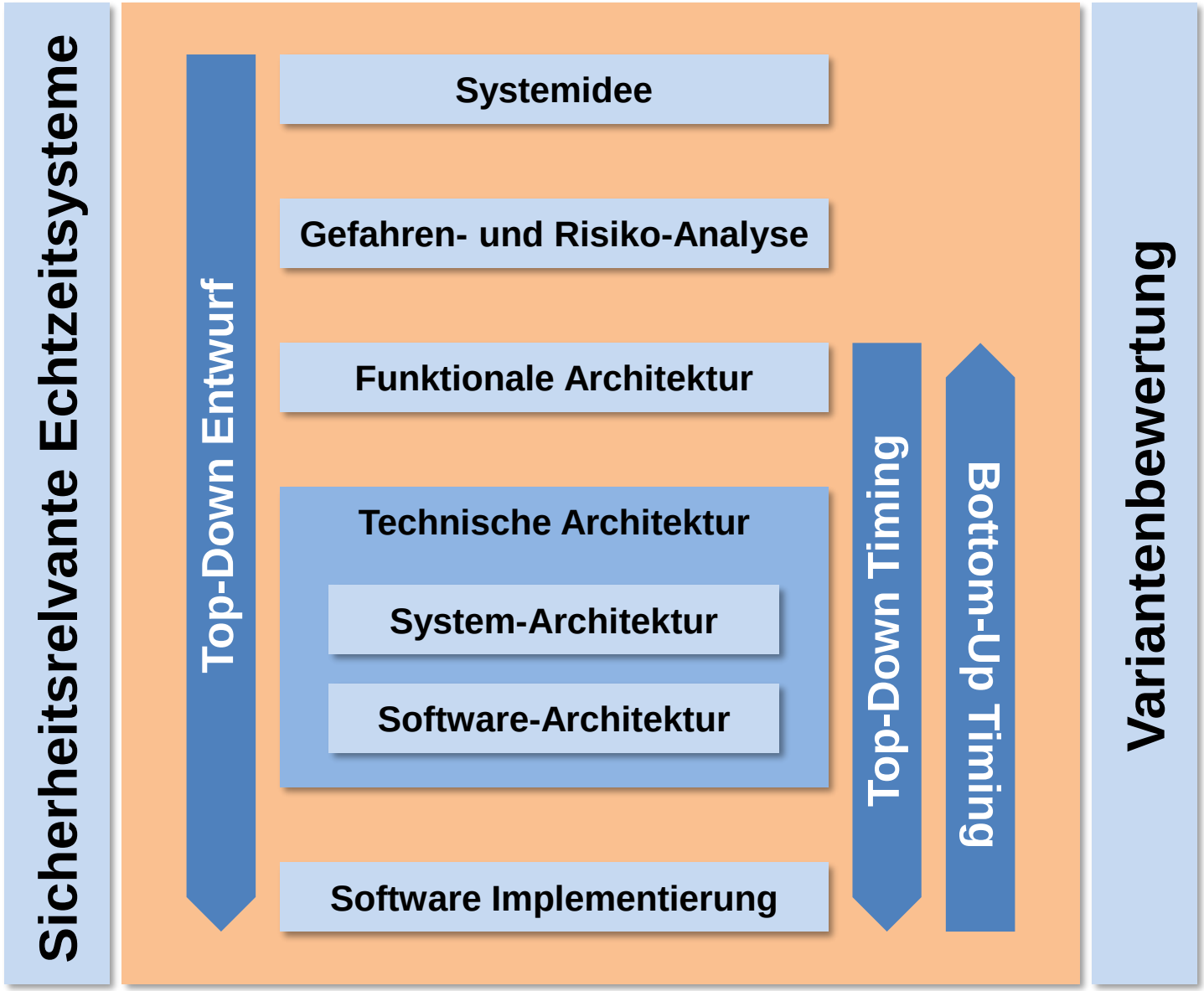
Das System sollte das Berechnungsergebnis bis zu einer festgelegten Deadline liefern. Falls dies nicht erreicht wird, verliert das Ergebnis an Wert, es hat aber keine schlimmen Auswirkungen.



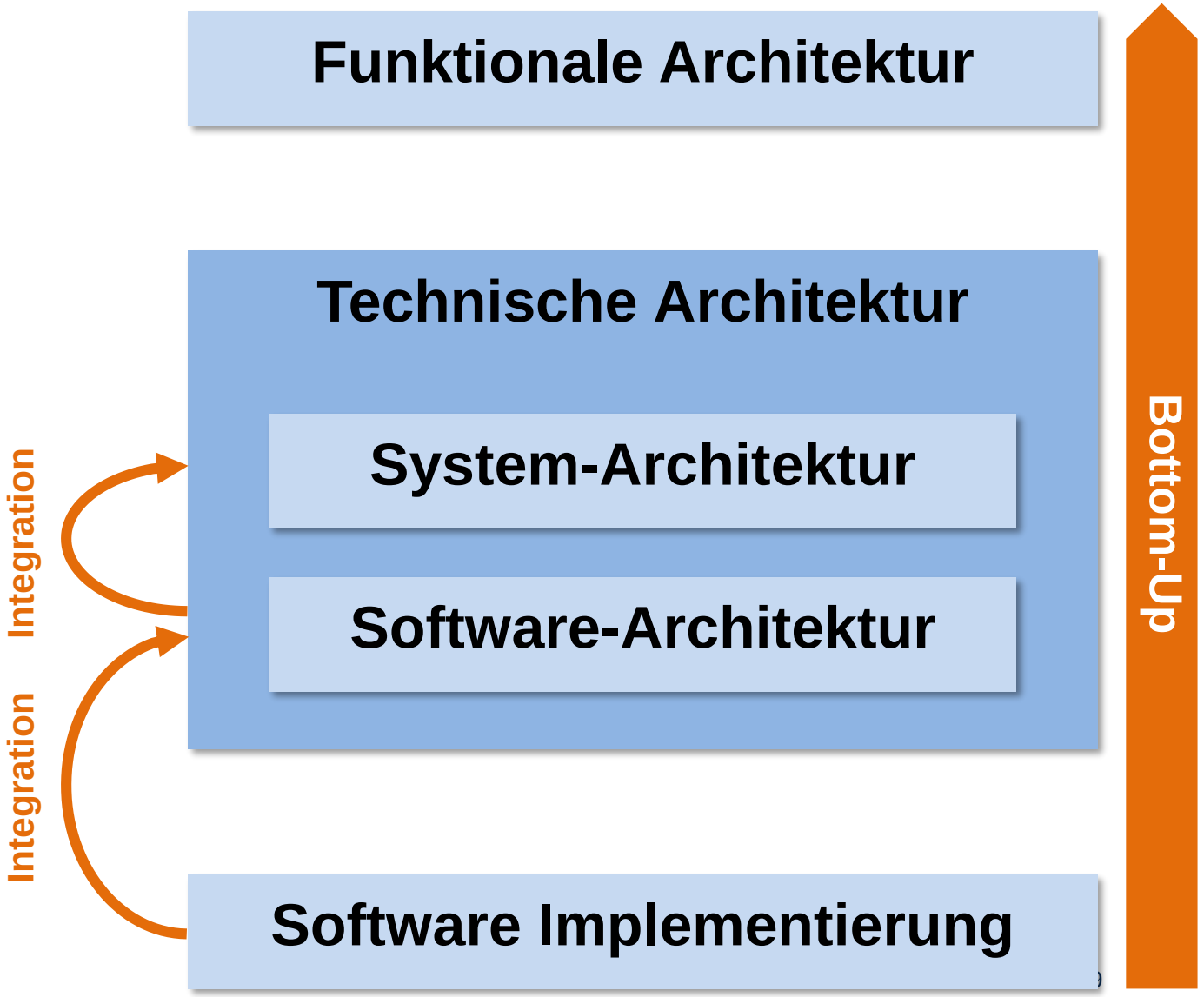
Hartes Echtzeitsystem

Das System muss das Berechnungsergebnis bis zur Deadline liefern. Falls dies nicht gewährleistet werden kann, muss eine benutzerdefinierte Ausnahmebehandlung eingeleitet werden.

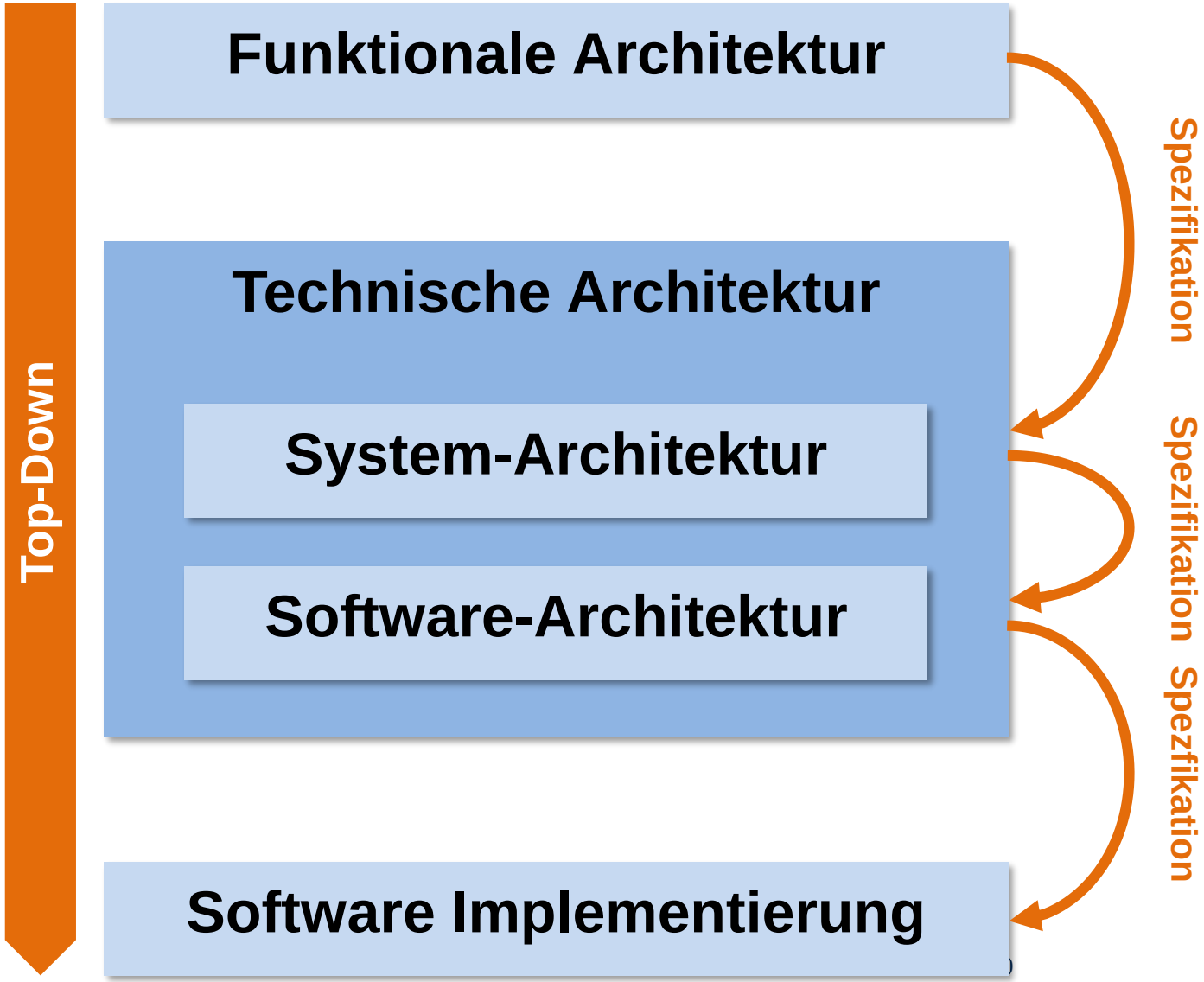
Überblick


Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

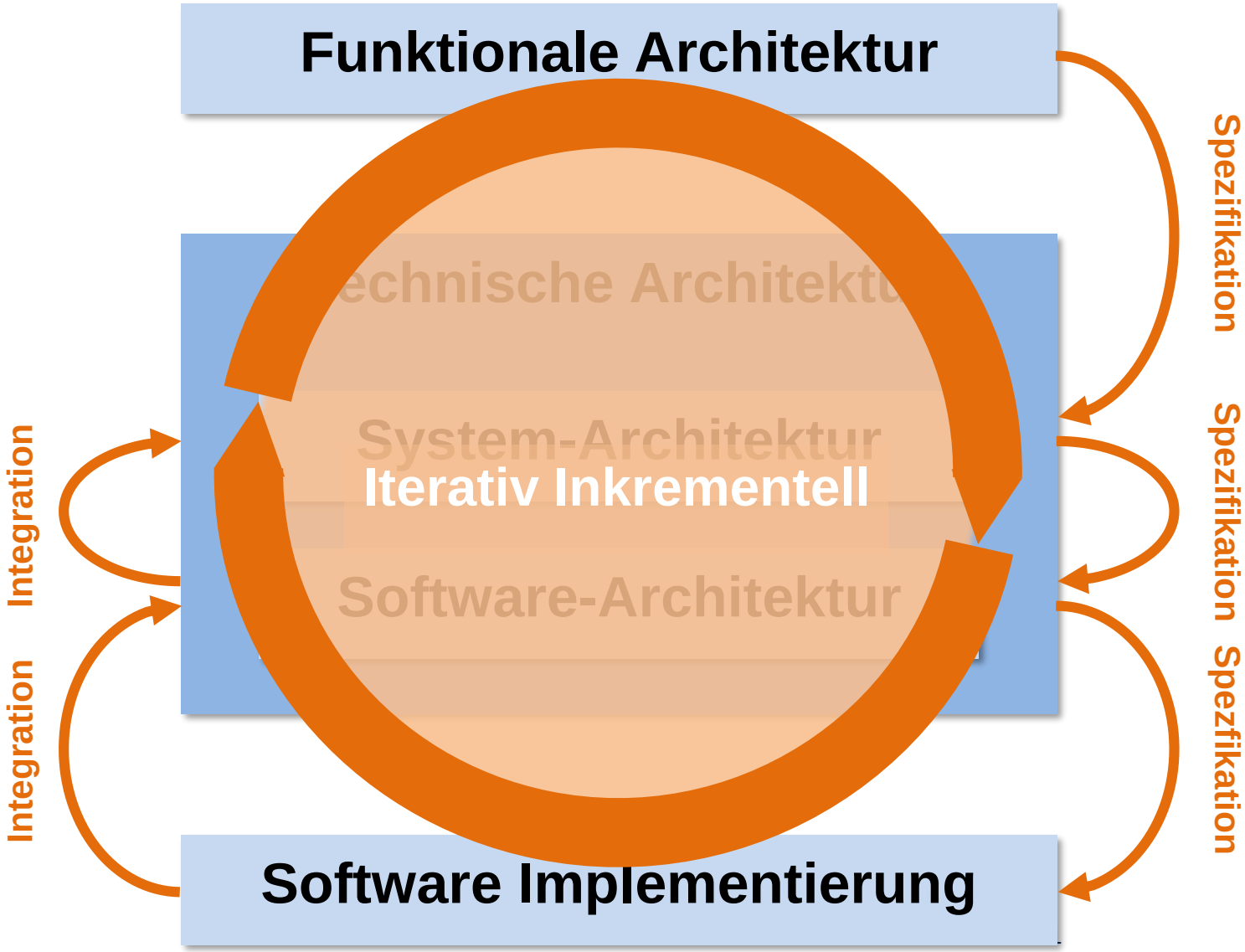
Wie konstruiert man ein Echtzeitsystem?



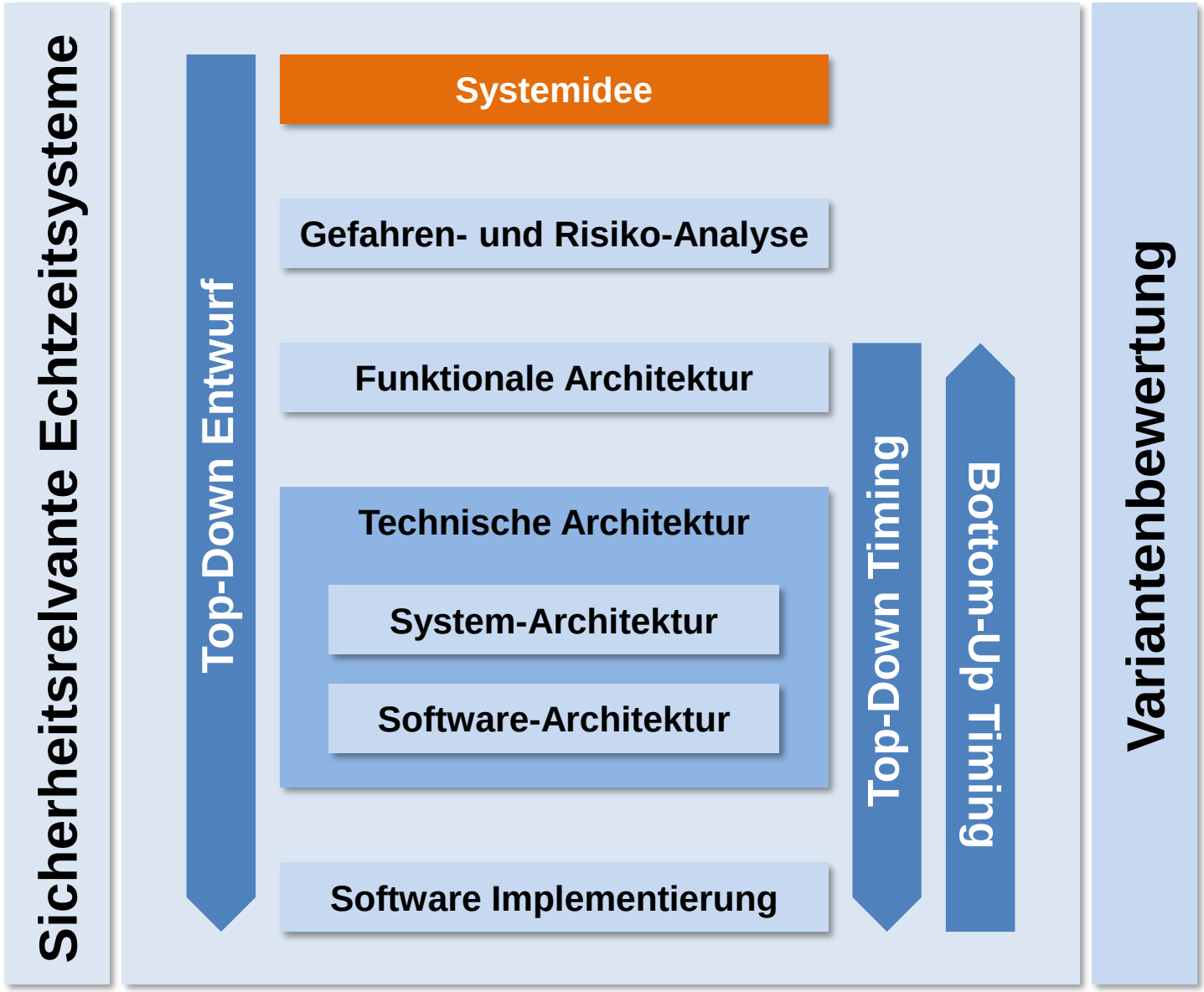
Wie konstruiert man ein Echtzeitsystem?



Wie konstruiert man ein Echtzeitsystem?

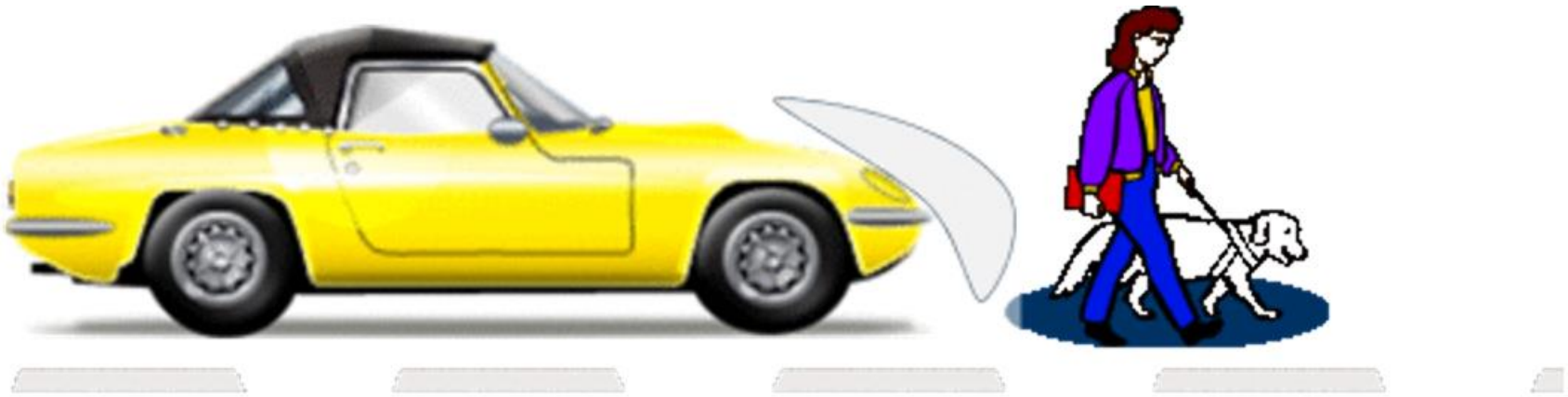


Überblick

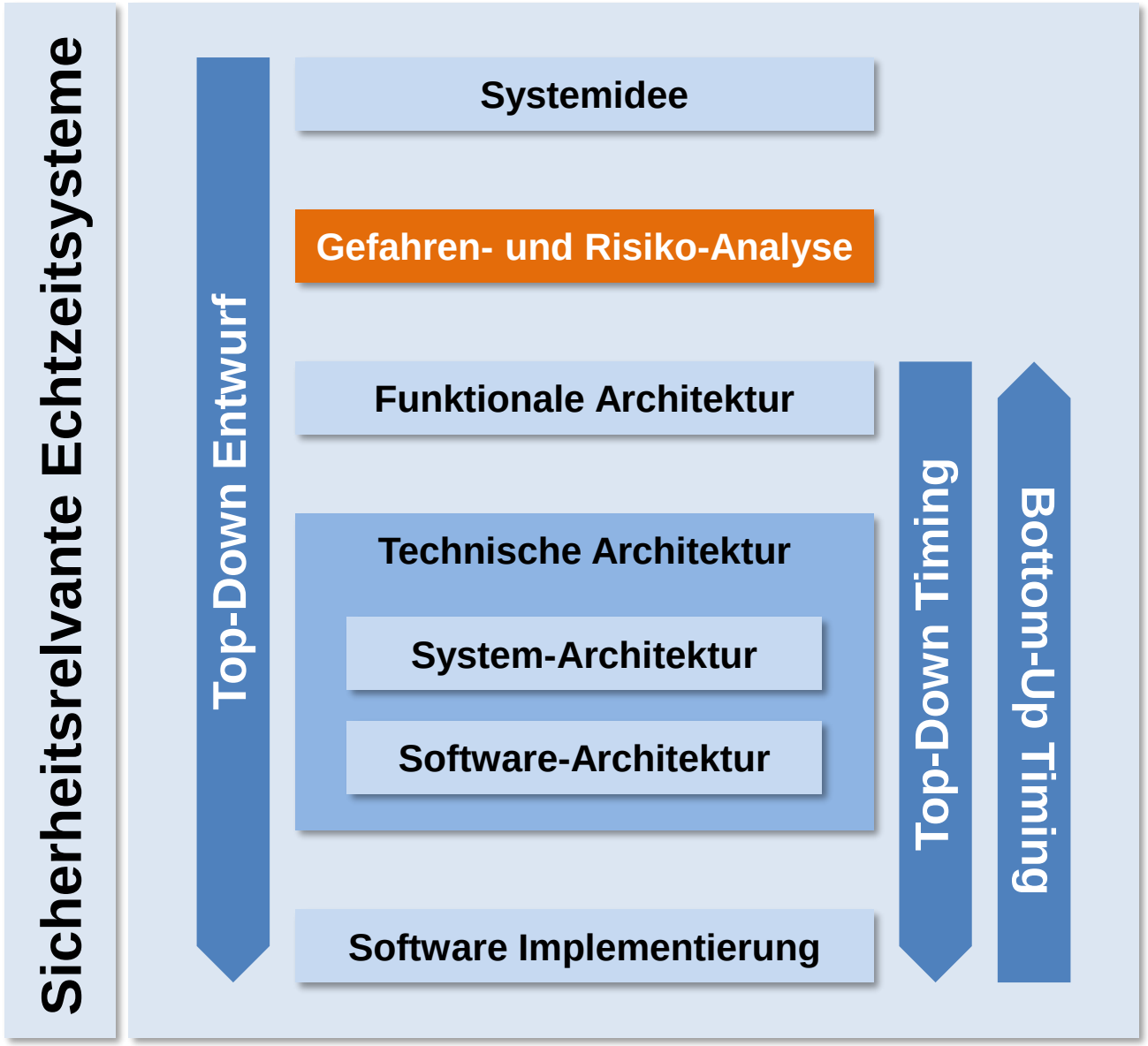

Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

Front Airbag System (FABSY)

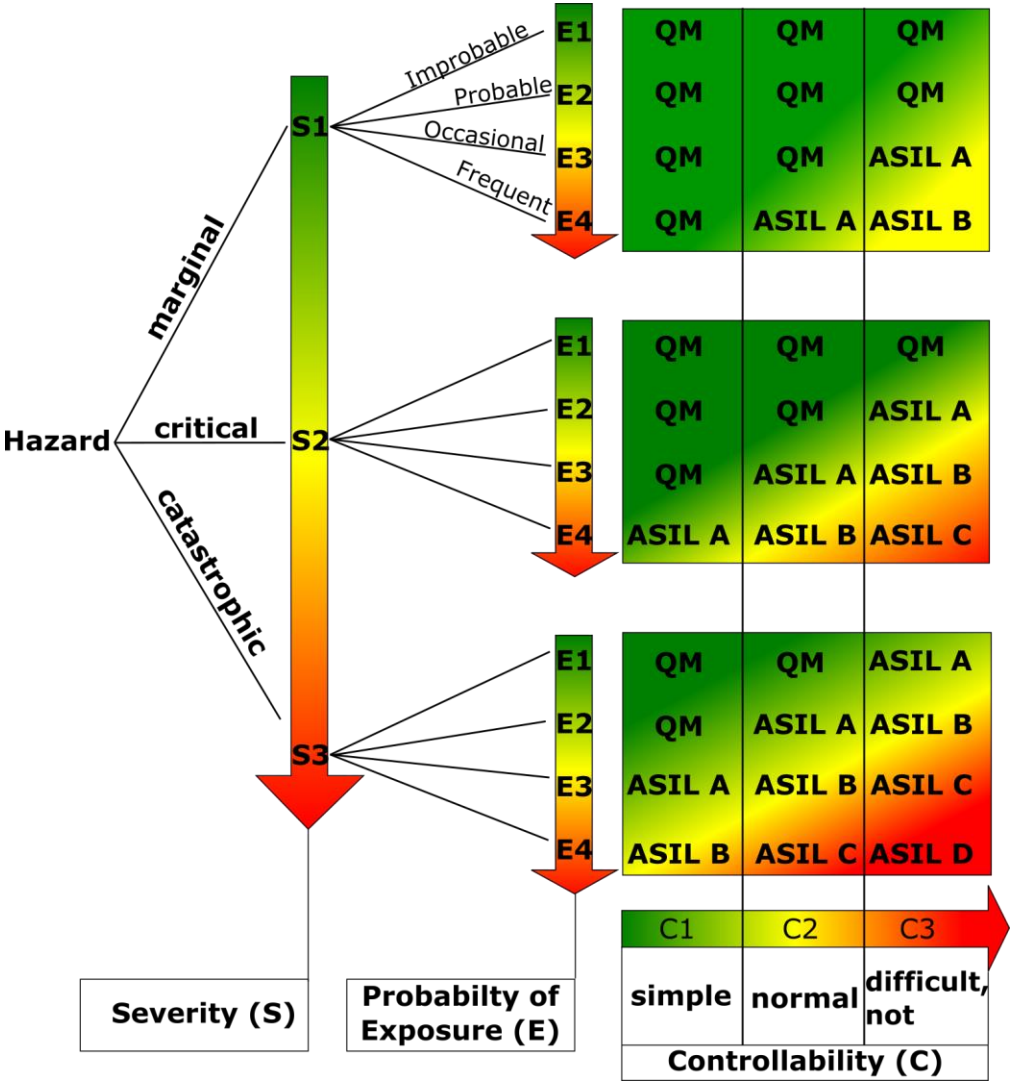
Ein Fahrzeug ist auf der Straße zügig unterwegs. Ein Fußgänger betritt plötzlich die Fahrbahn und eine Kollision ist unvermeidbar. Das FABSY-Steuergerät erkennt den Fußgänger und löst den Airbag aus, um schweren Schaden vom Fußgänger abzuwenden. Es handelt sich um ein **hartes Echtzeitsystem**!



Überblick


Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing

Gefahren- und Risikoanalyse



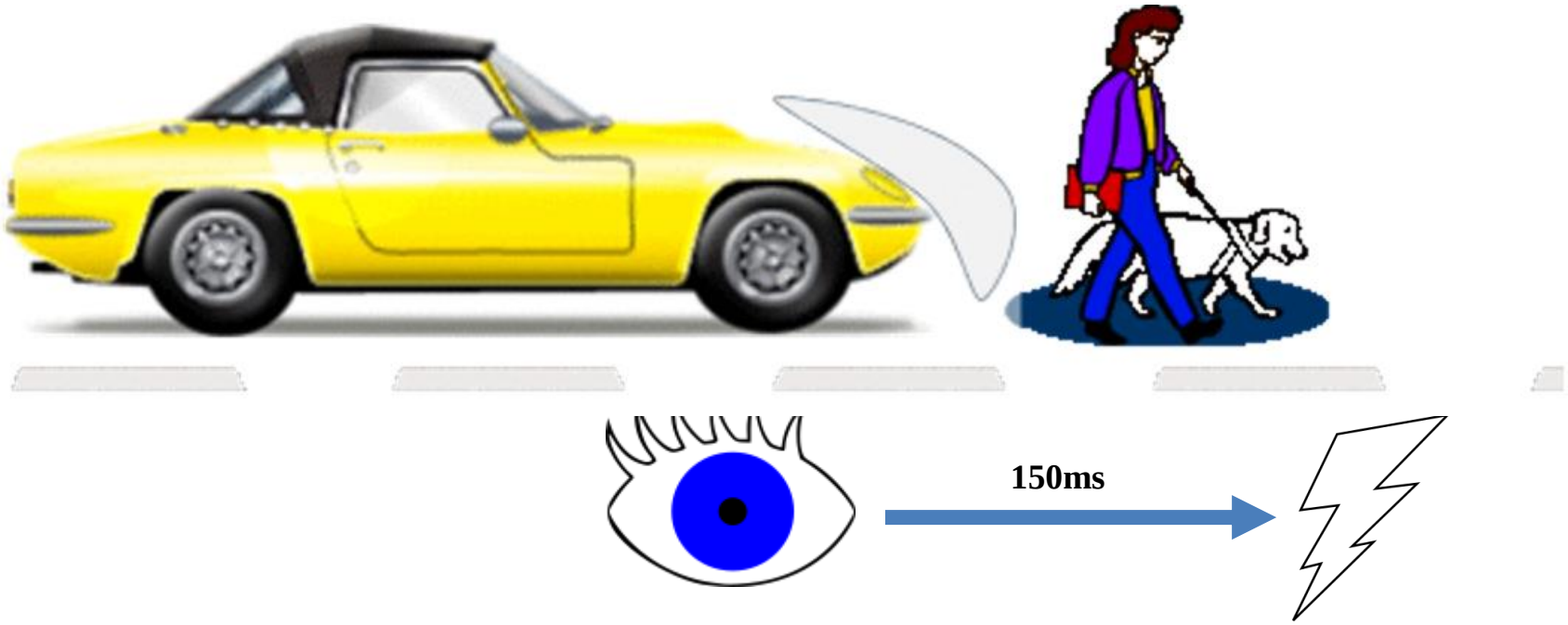
Gefahren- und Risikoanalyse für FABSY

ID	Scenario considered	Mal function	Effect of failure	S		Comment for Severity	E		Comment for Exposure Time	C		Comment for Controllability	ASIL
B2	Fast driving within the city, pedestrian walks onto the road	Airbag does not inflate, false negative	Pedestrian collides with car	S3	Life-threatening injuries (survival uncertain), fatal injuries	life threatening or fatal injuries	E2	Low probability	pedestrians walking onto highway / freeway is possible but infrequent	C3	Difficult to controll or uncontrollable by skilled driver	situation cannot be controlled	B
B3	Slow driving within the city, pedestrian walks onto the road	Airbag does not inflate, false negative	Pedestrian collides with car	S2	Severe and life threatening injuries (survival probable)	severe and life threatening injuries	E2	Low probability	pedestrians walking onto highway / freeway is possible but infrequent	C2	Controllable by skilled driver	situation is controllable by driver with training	QM
B5	Lane change on highway or freeway, strong traffic	Airbag inflates, false positive	Driver is shocked, looses control over car	S3	Life-threatening injuries (survival uncertain), fatal injuries	life threatening or fatal injuries	E4	High probability	Lane change on highway is very frequent	C3	Difficult to controll or uncontrollable by skilled driver	situation is difficult to control for a skilled driver	D

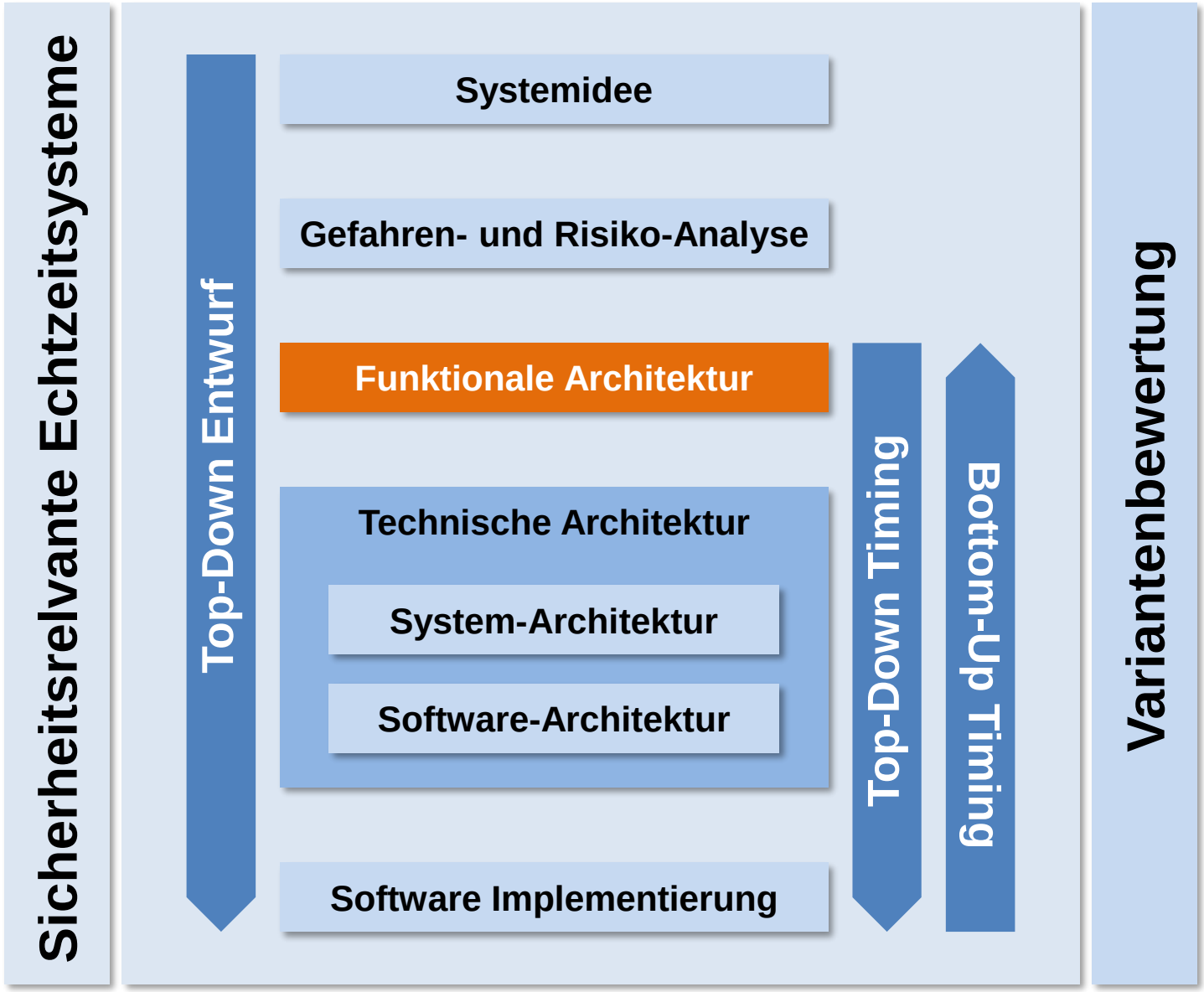
Sicherheitsziele und Zeitanforderungen

<u>Sicherheitsziel 1</u>: Verhindere ungewolltes Auslösen des Airbags (**ASIL D**).

<u>Sicherheitsziel 2</u>: Wenn eine Kollision mit einem Fußgänger unvermeidbar ist, garantiere das Auslösen des Airbags (**ASIL B**) zu einem **bestimmten Zeitpunkt**.



Überblick

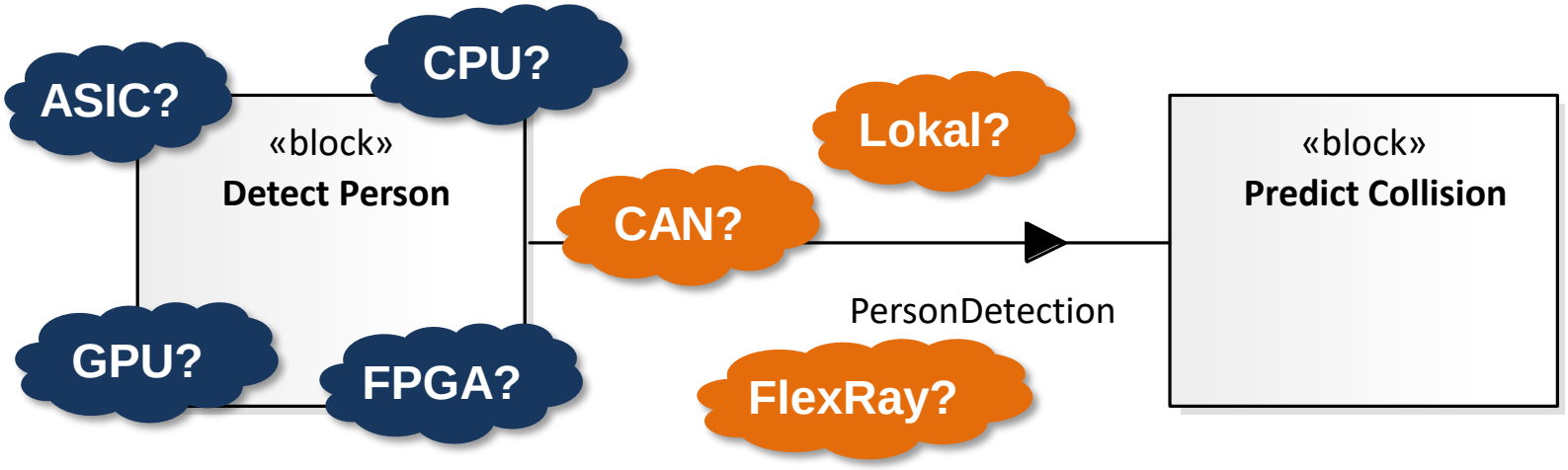

Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

Funktionale Architektur

Betrachtung des Systems aus rein funktionaler Sicht:

- Zerlegung in Funktionsblöcke
- Zusammenwirken der Funktionsblöcke durch Informationsflüsse

Modellierung der System-Funktionalität unabhängig von der konkreten technischen Realisierung



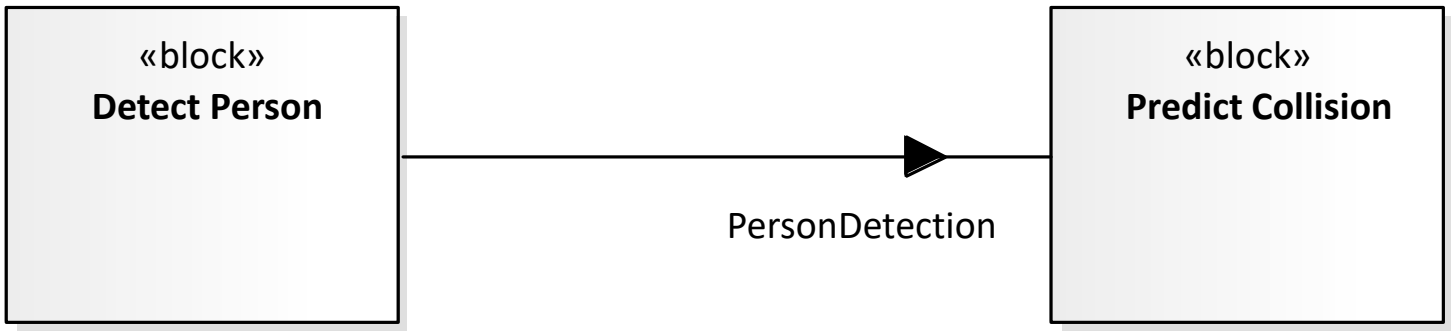
Funktionale Architektur – wozu?

→ Basis für Ableitung von Wirkketten

→ Basis für Analyse und Simulation der Funktion

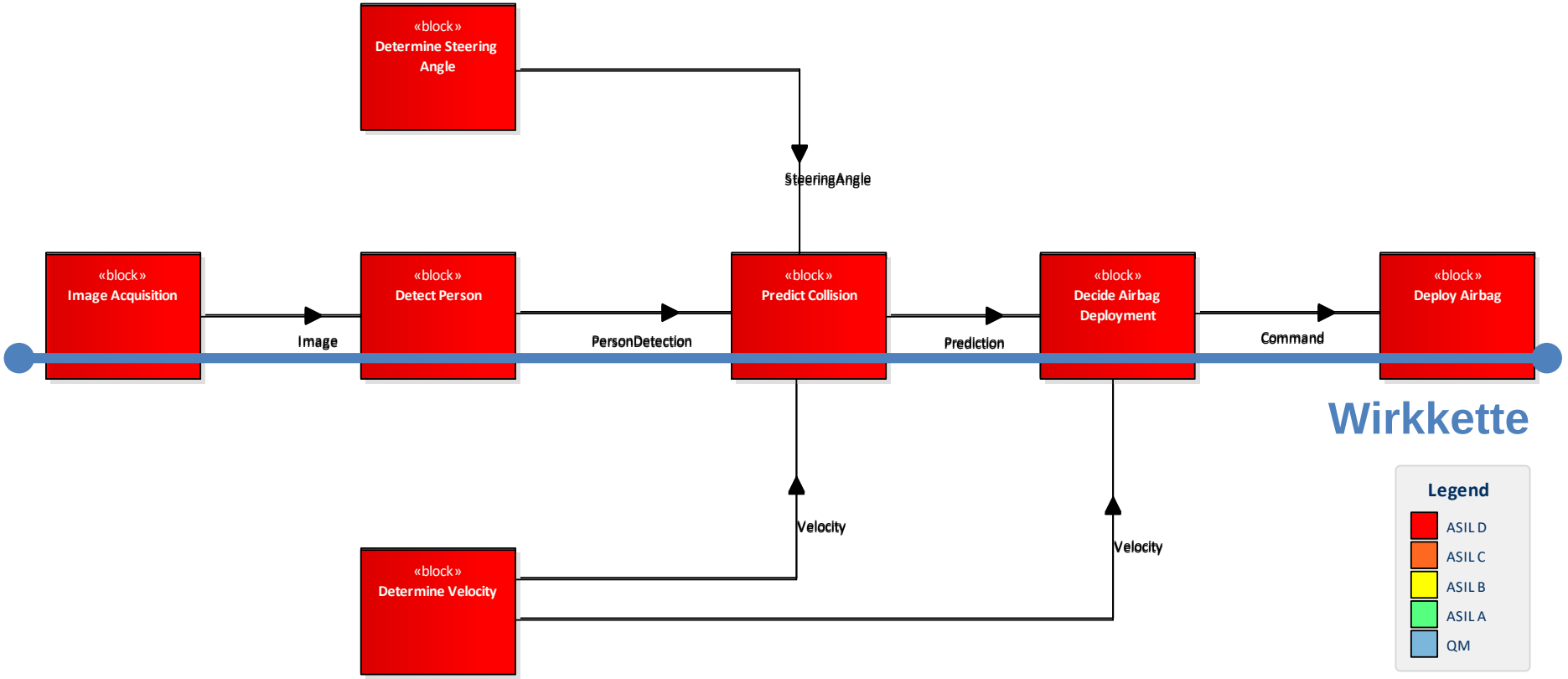
→ Freiheitsgrade für die technische Realisierung

→ Gültig für verschiedene technische Architekturvarianten

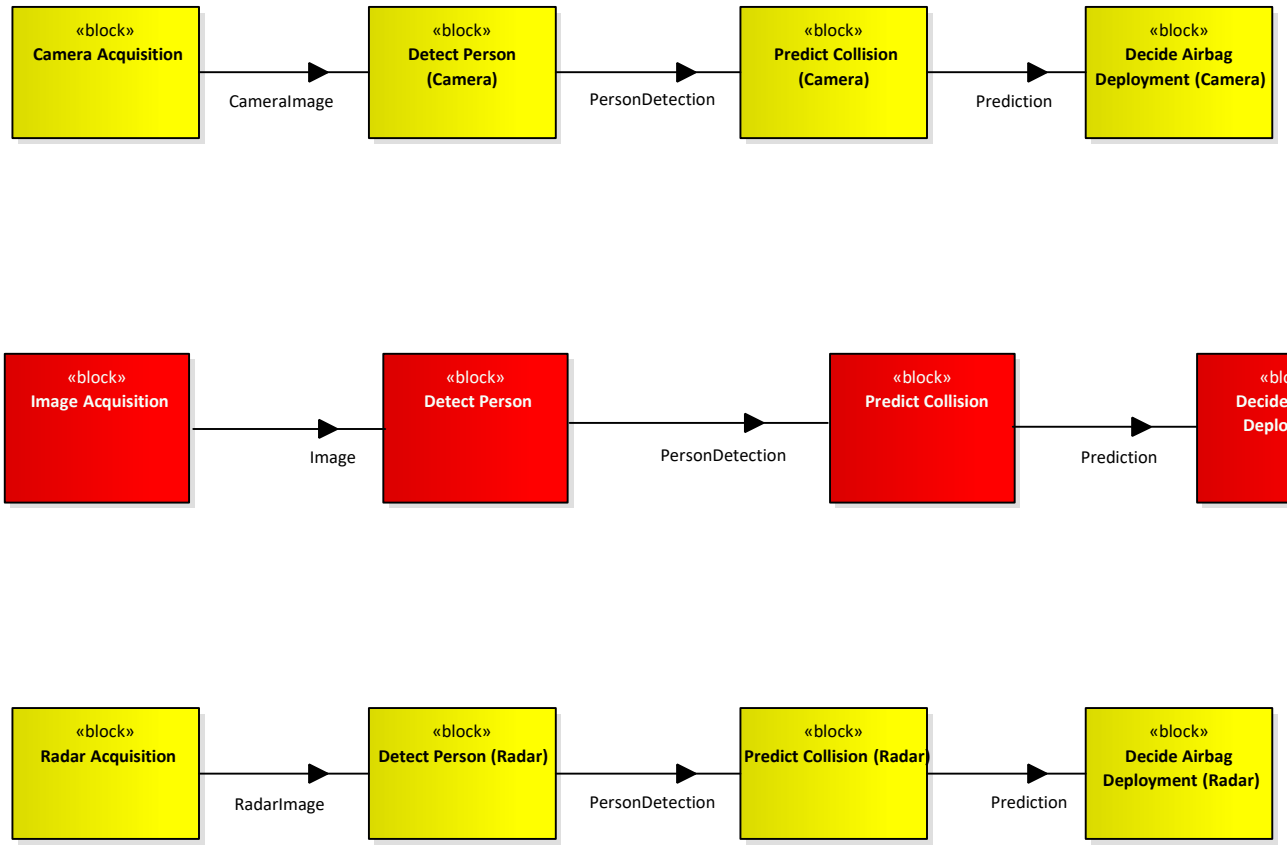


Funktionale Architektur – FABSY

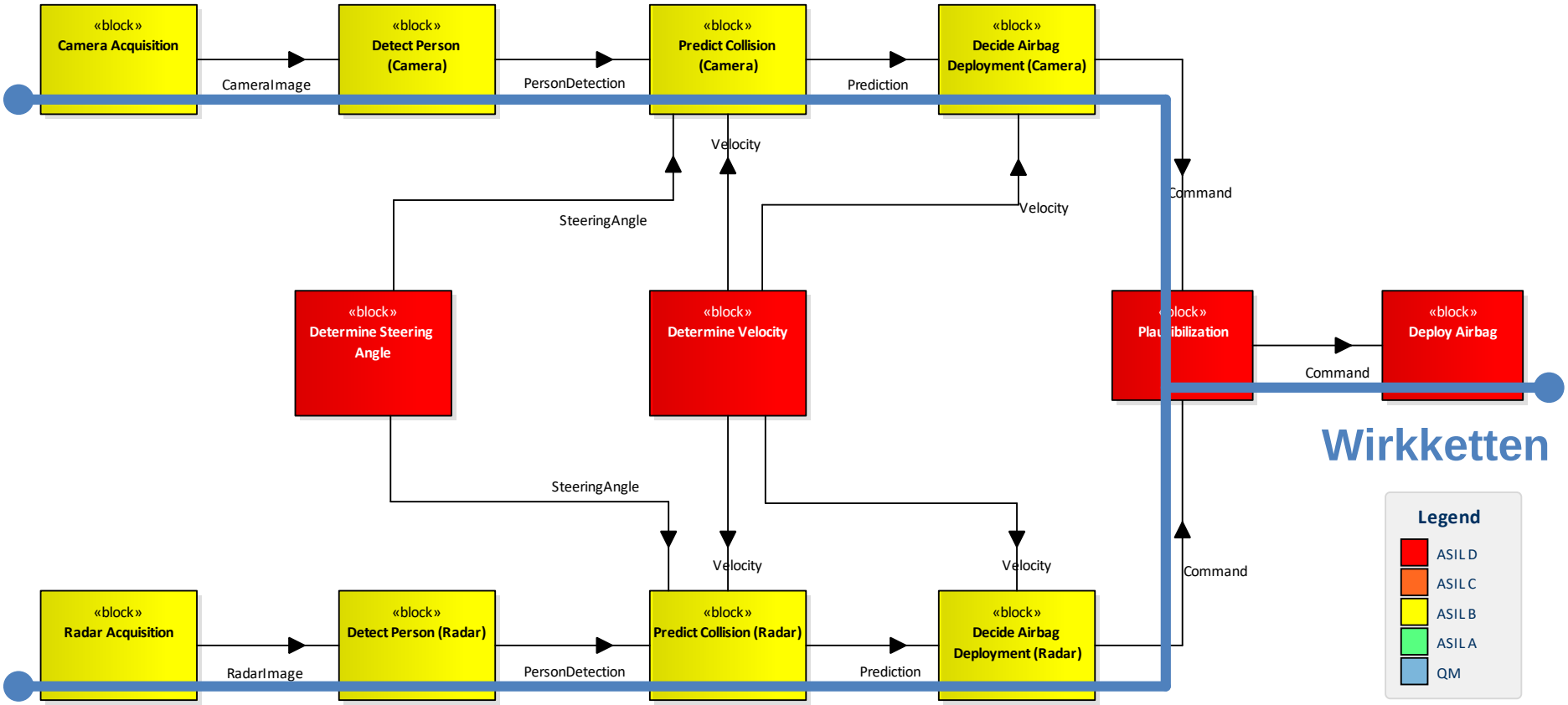
Sicherheitsziel 1: Verhindere ungewolltes Auslösen des Airbags (ASIL D).



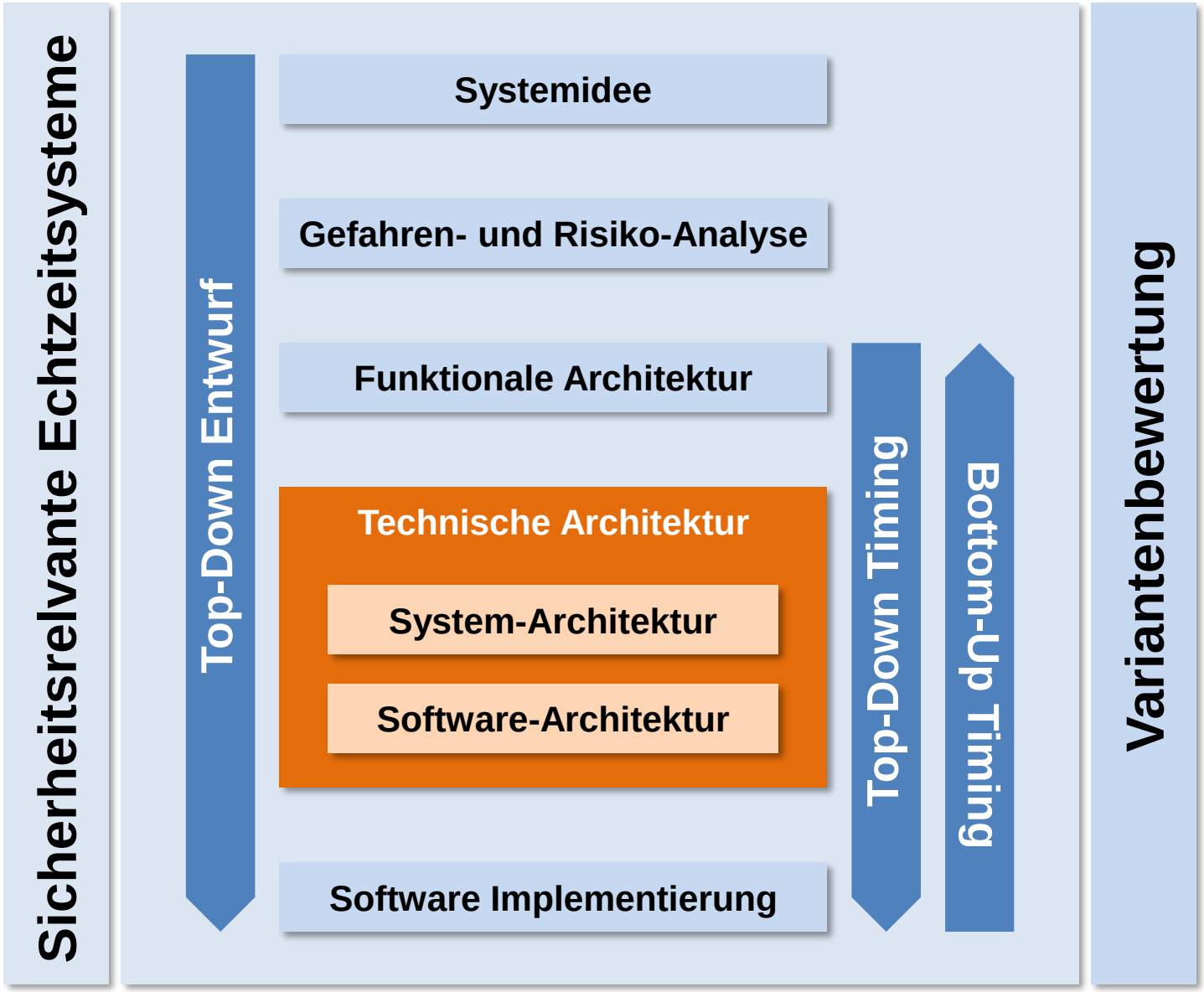
Funktionale Architektur – FABSY ASIL Dekomposition



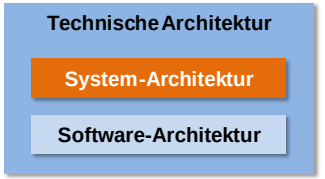
Funktionale Architektur – FABSY ASIL Dekomposition



Überblick


Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

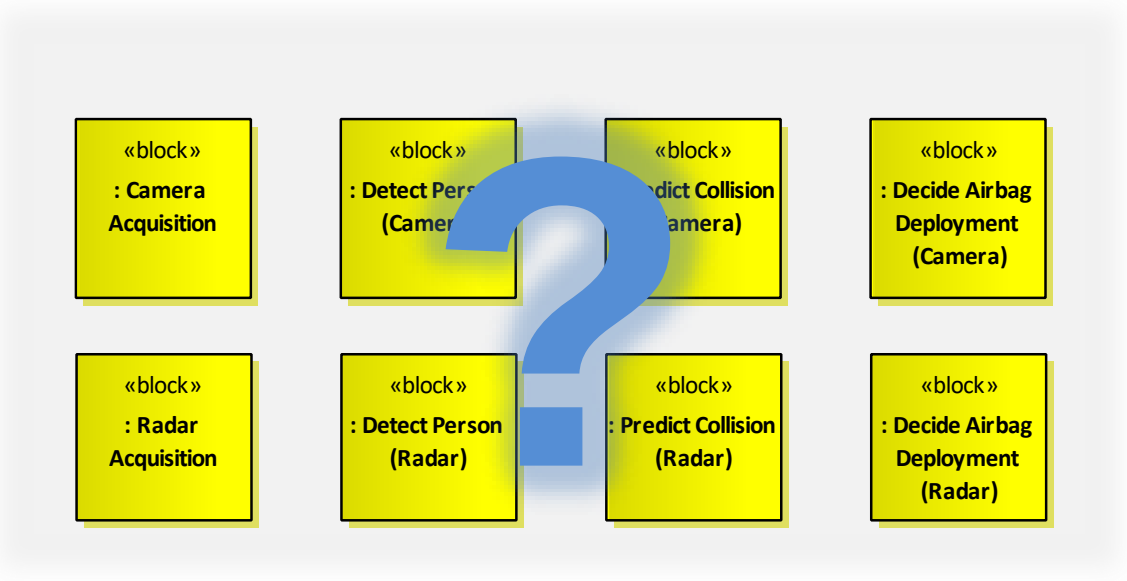
Technische Architektur

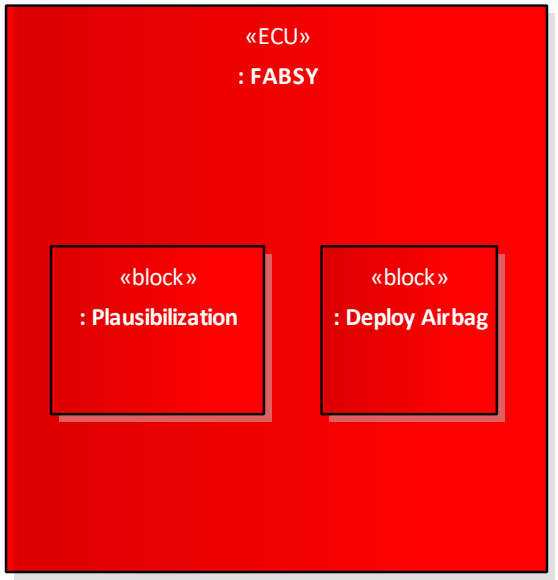


System-Architektur

- Realisierung der Funktionsblöcke:
  Hardware, Software, Mechanik, Kombination…
- Allokation der Funktionsblöcke auf konkrete Systemkomponenten
  (realisiert in Hardware, Software, Mechanik…)
- Definition der Steuergeräte und ihrer Vernetzung
- Verteilung der Systemkomponenten auf die Steuergeräte

Technische Architektur für FABSY

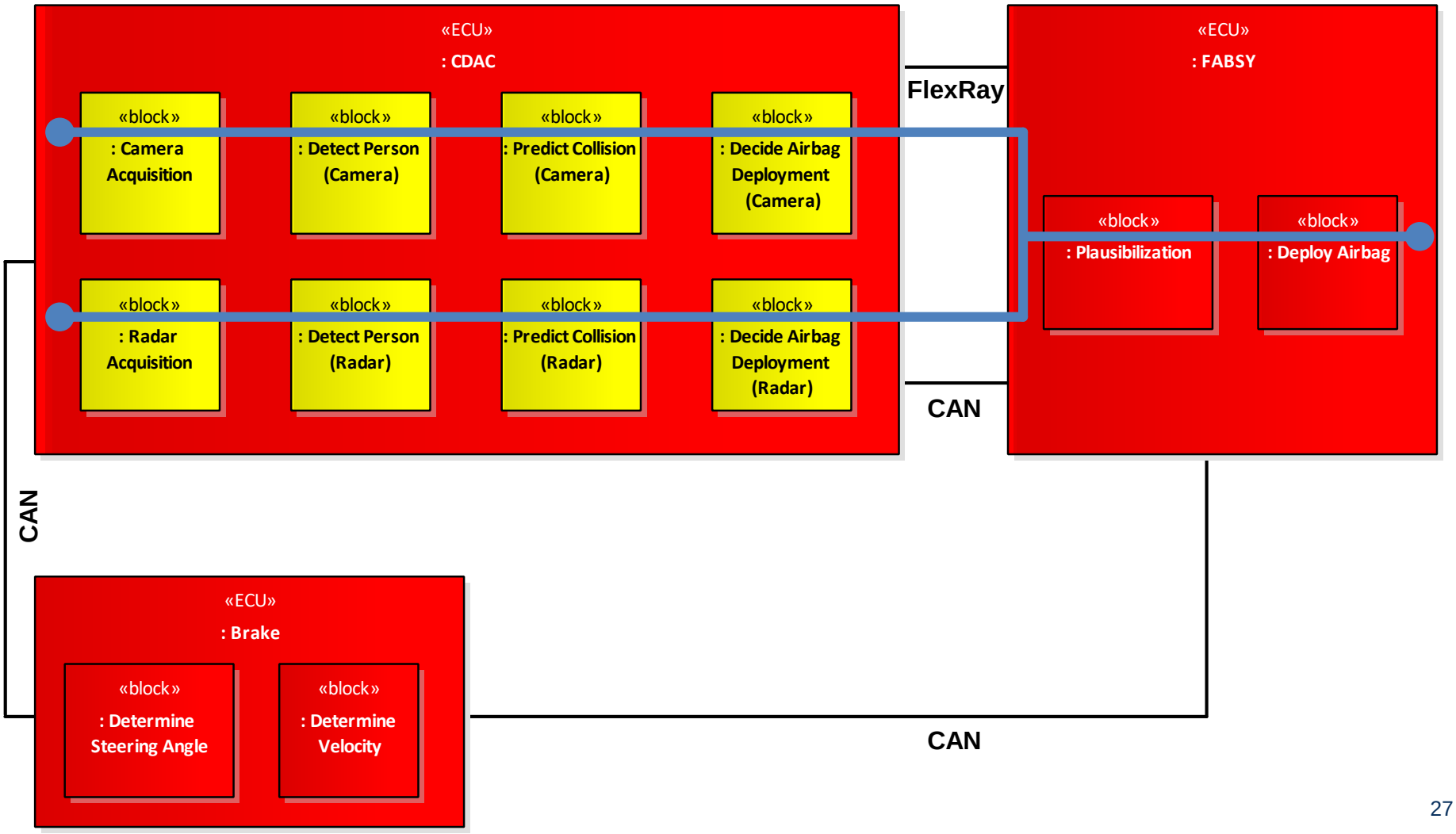




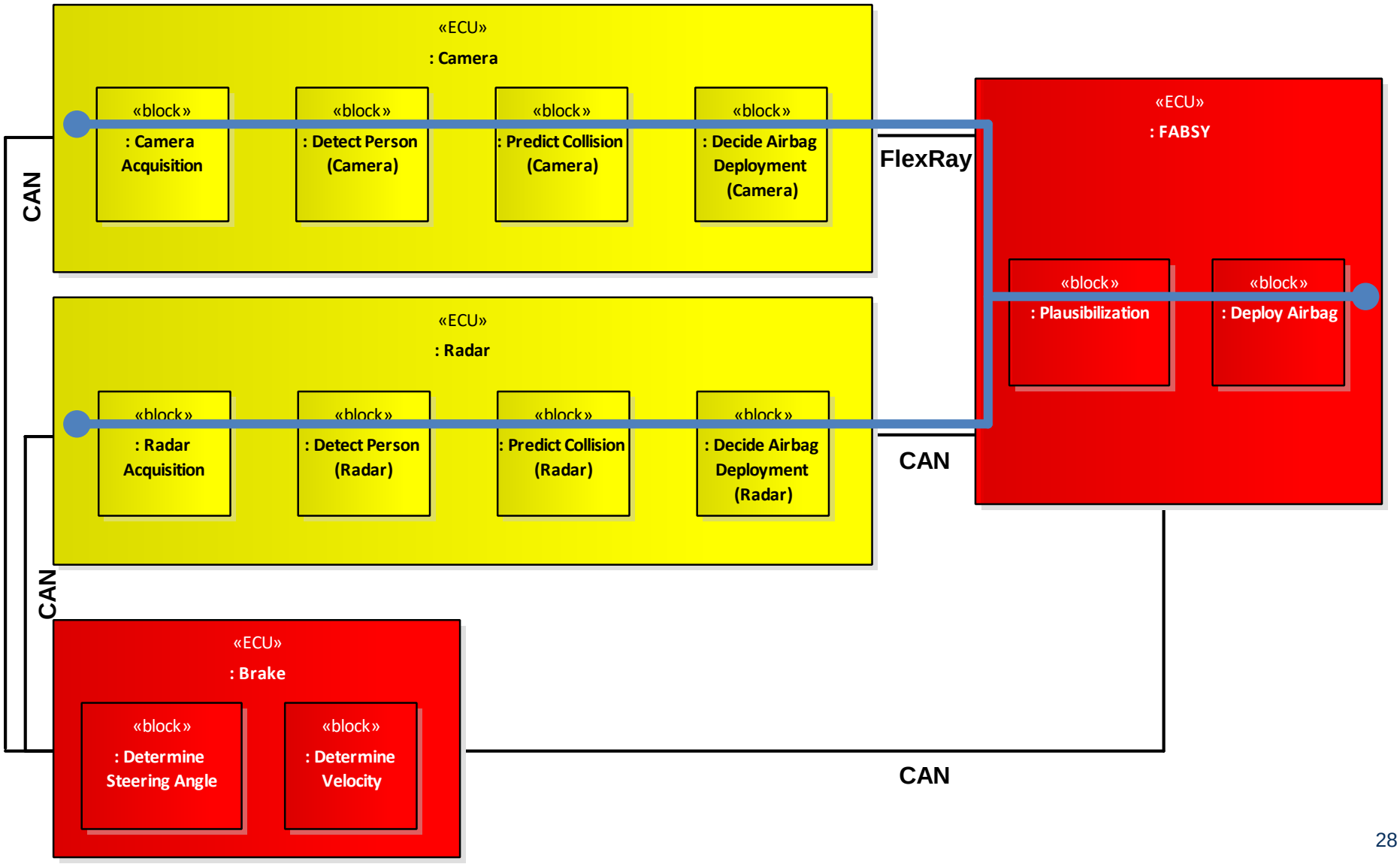
Variante 1: Integrierte Architektur mit CDAC ECU ("Central Driver Assistance Controller")

Variante 2: Föderierte Architektur mit dedizierten Steuergeräten "Camera ECU" und "Radar ECU"

Technische Architektur für FABSY: CDAC ECU



Technische Architektur für FABSY: Camera + Radar ECU



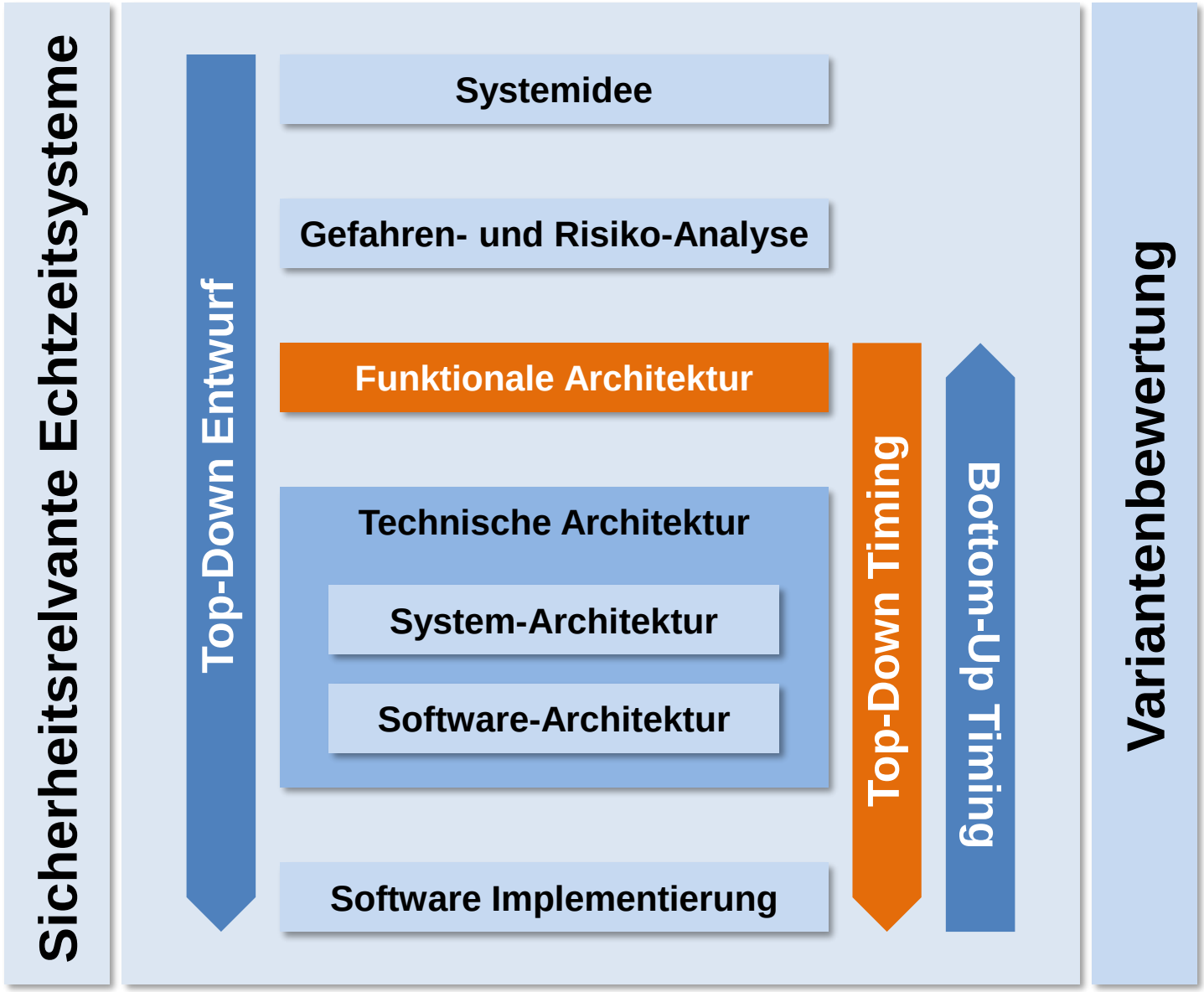
Technische Architektur

Software-Architektur

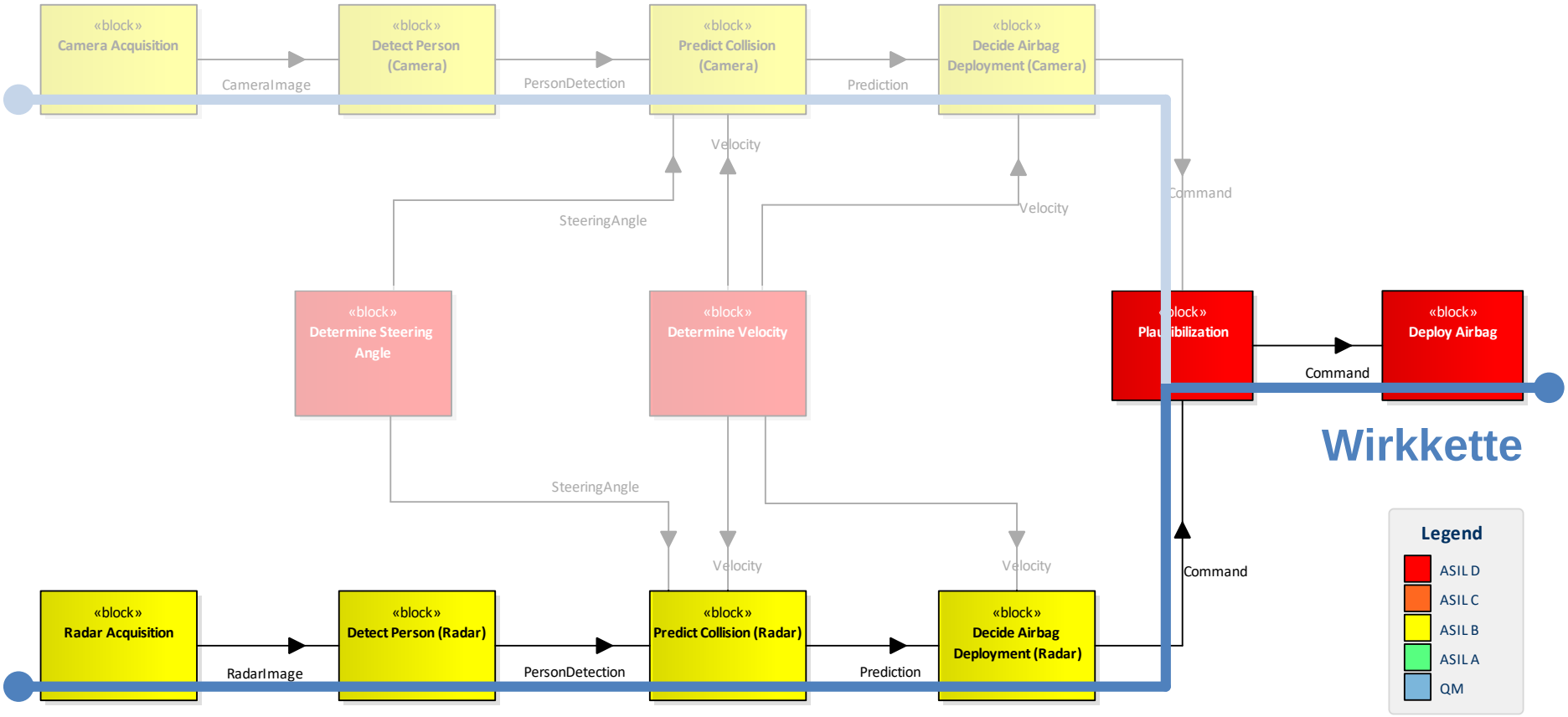
Zerlegung der Software-basierten Systemkomponenten:

- Definition von Software-Komponenten mit ihren Verantwortlichkeiten und Schnittstellen
- Auswahl von Technologien, Mustern, …
- Allokation von Software-Funktionen auf Tasks und Interrupts
- Definition von Priorität und Periode für Tasks und Interrupts
- Allokation von Tasks und Interrupts auf Cores

Überblick

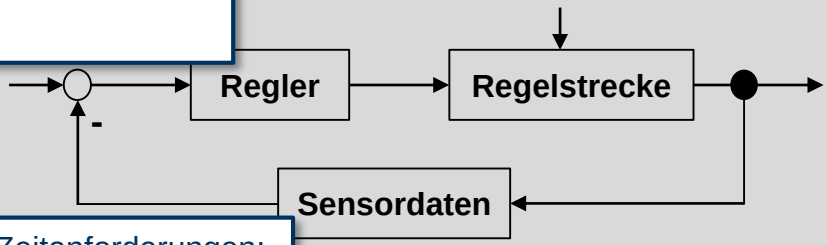

Sicherheitsrelevante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

Funktionale Architektur – FABSY, mit ASIL Dekomposition



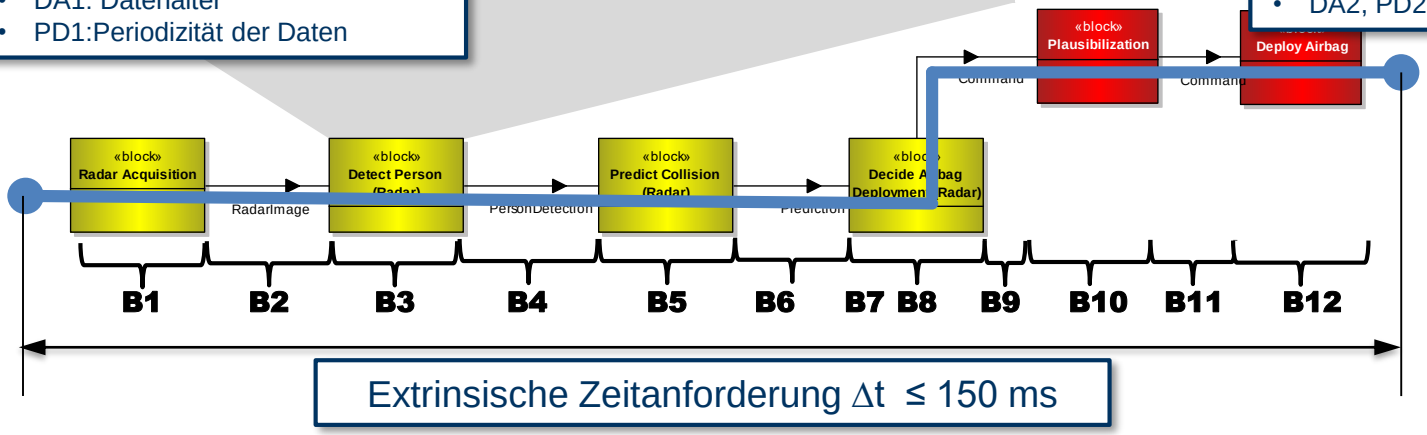
Funktionale Architektur: Identifikation und Budgetierung der Wirkketten

Intrinsische Zeitanforderungen:
- P1: Periodizität
- J1: Jitter



Intrinsische Zeitanforderungen:
- DA1: Datenalter
- PD1:Periodizität der Daten

Intrinsische Zeitanforderungen:
- P2, J2
- DA2, PD2

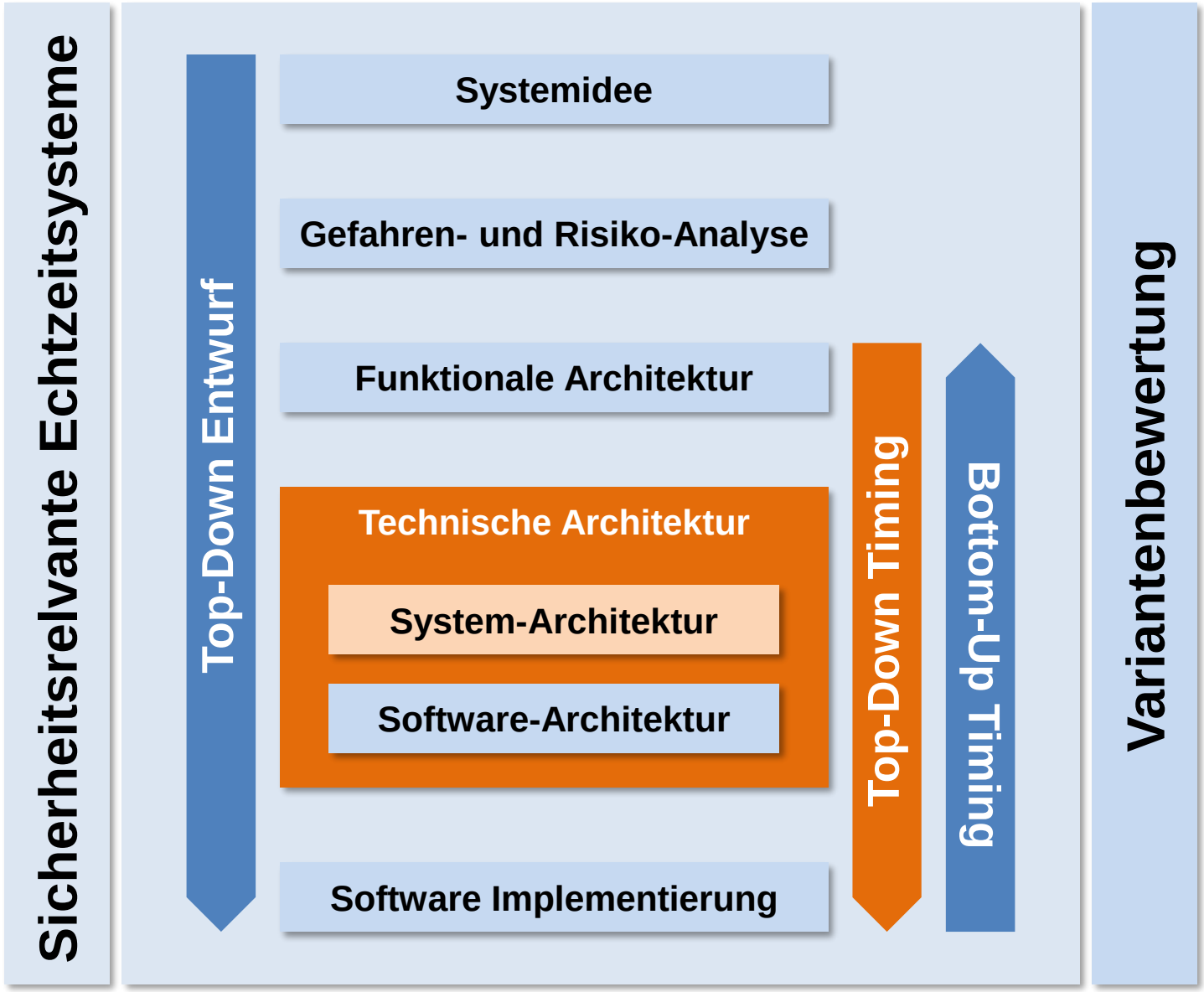


Budgetierung Funktionsblöcke und Kommunikation

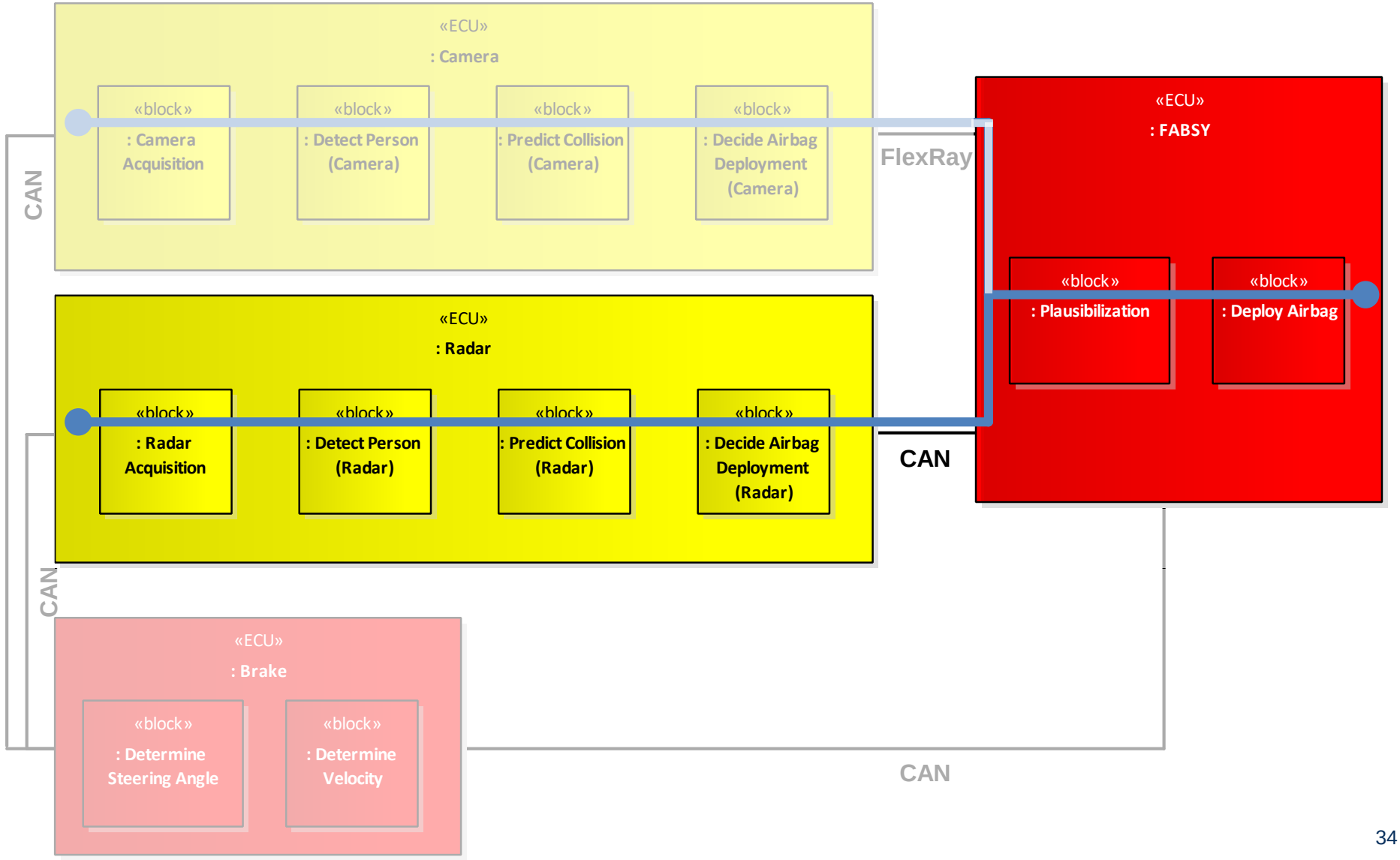
Modellbasierte Absicherung durch **Simulation** und/oder **Worst-Case Analyse**

- Extrinsische Zeitanforderungen: Ende-to-End Latenzen Wirkketten
- Intrinsische Zeitanforderungen: Periodizität, Jitter, Datenalter, …

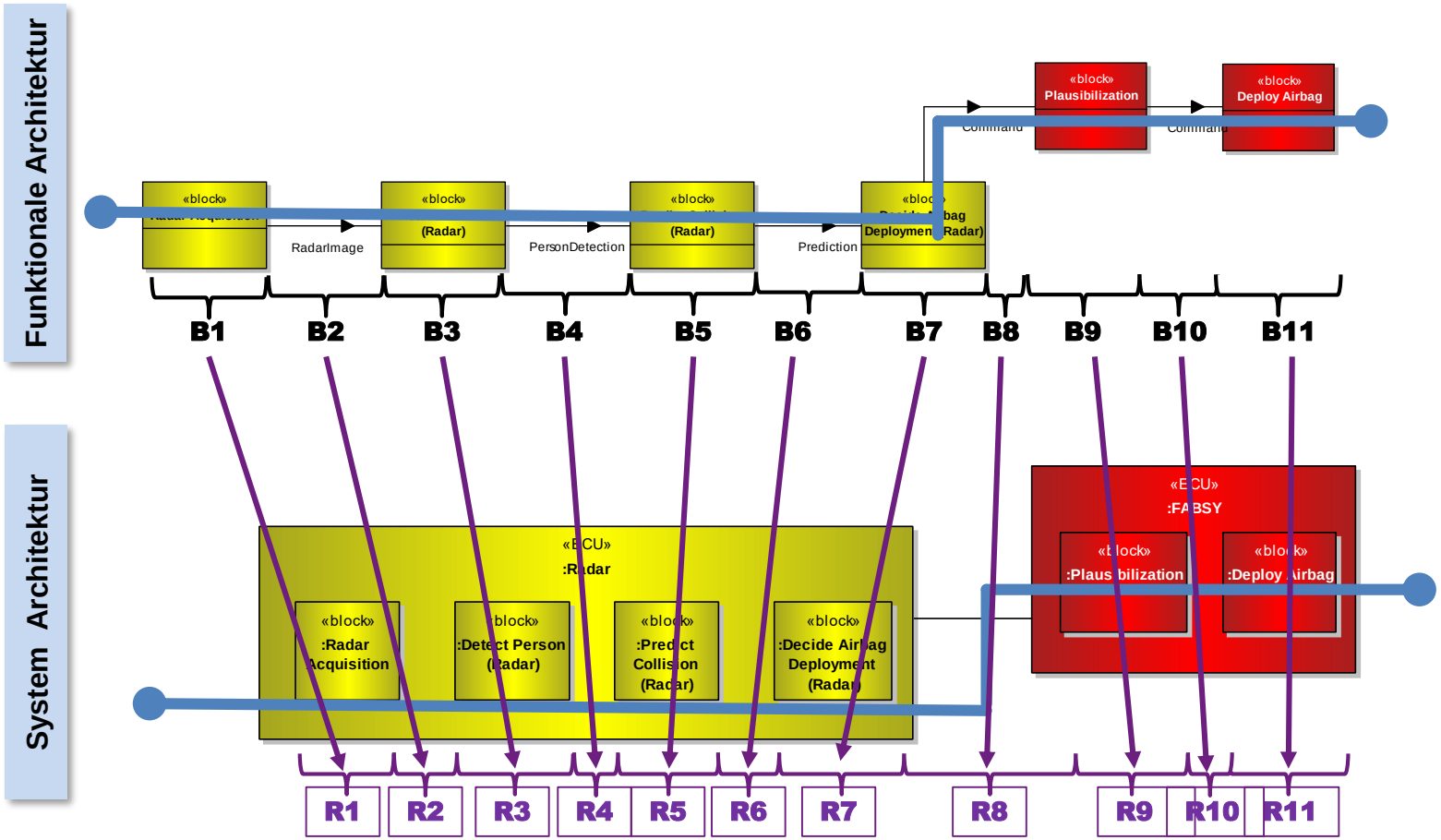
Überblick


Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

Technische Architektur für FABSY: Camera + Radar ECU

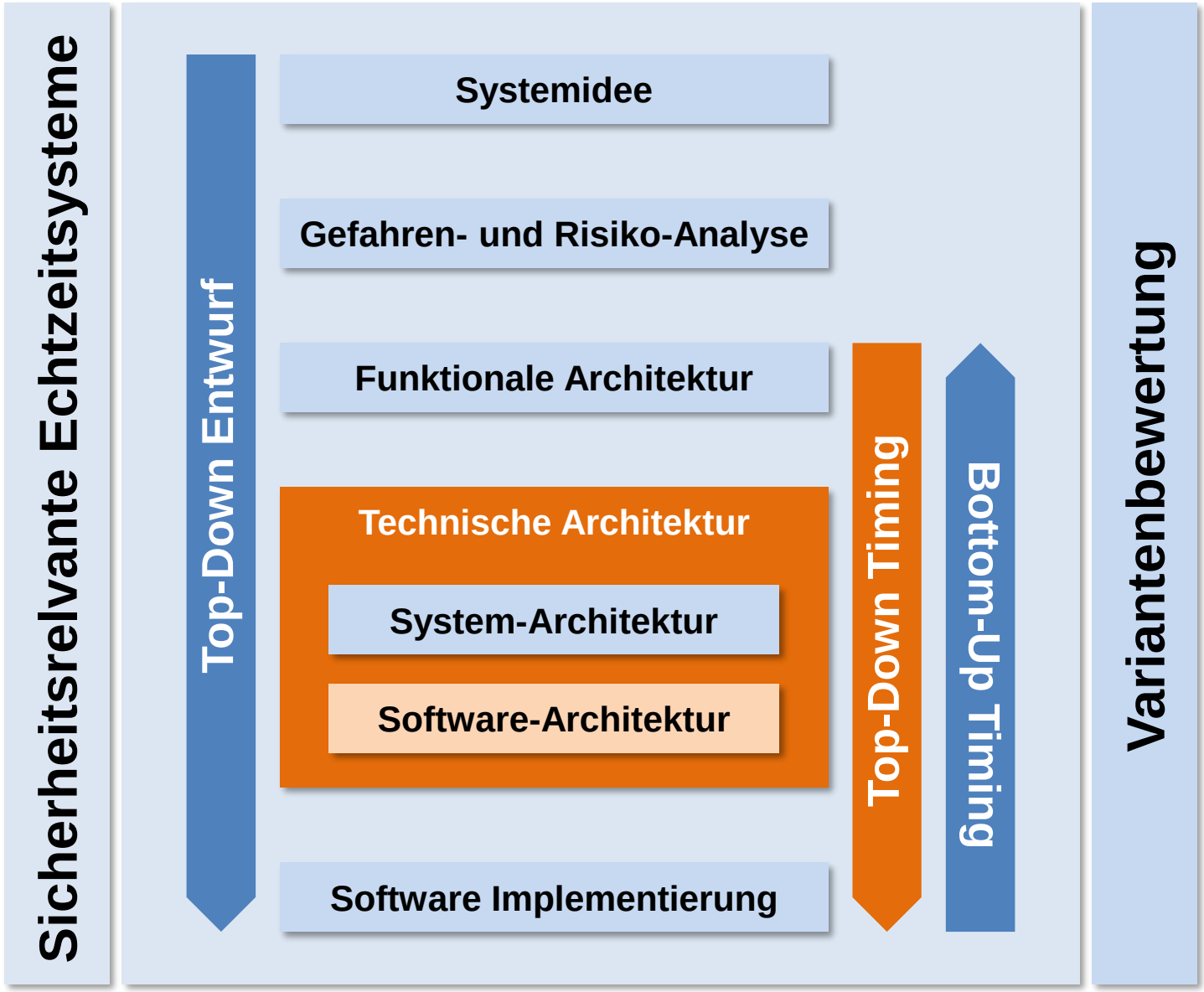


Requirements für Technische Architektur Camera + Radar ECU



Aus **Budgets** in der Funktionalen Architektur **werden Requirements** für die System Architektur.

Überblick

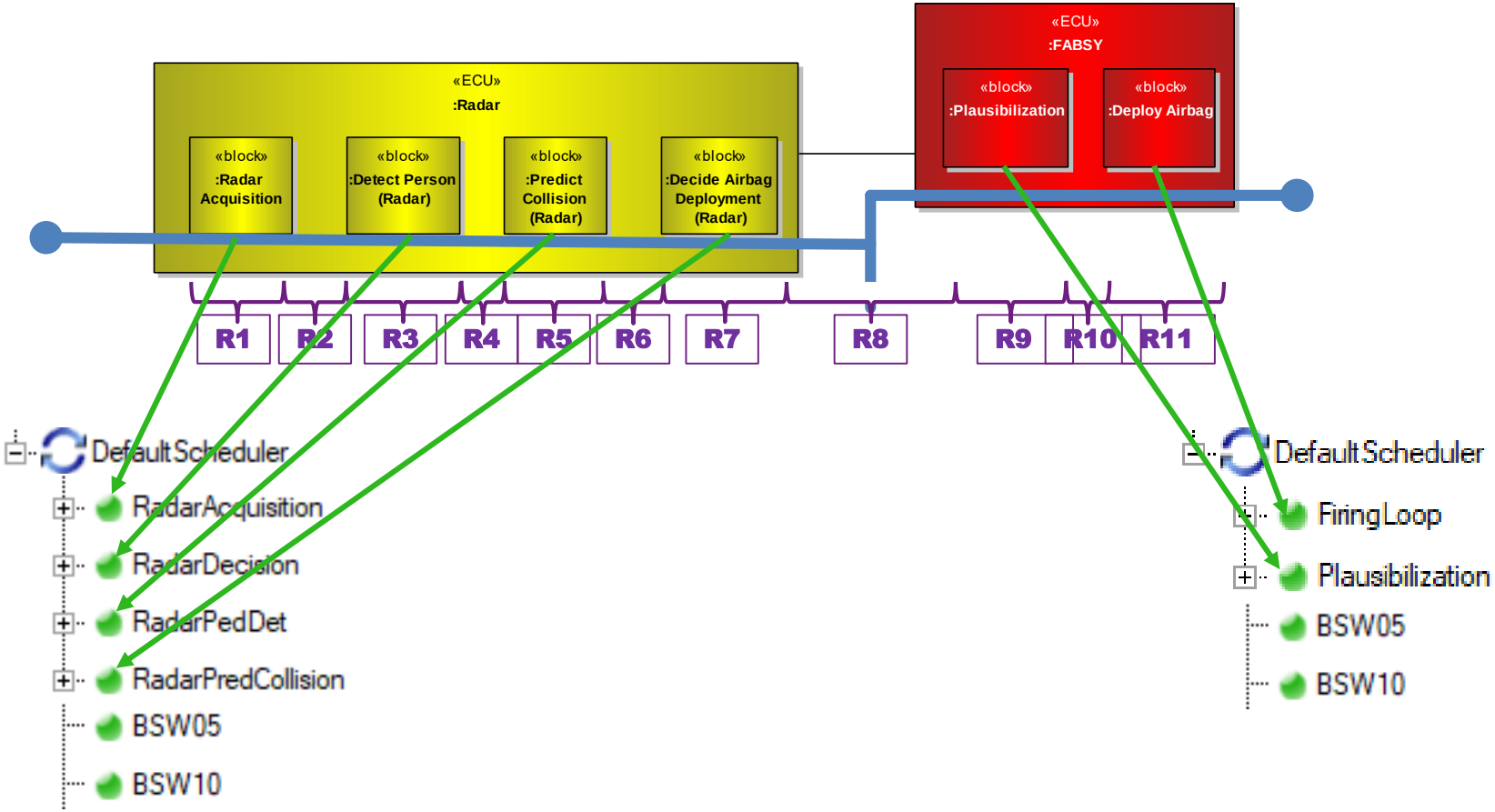

Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

Ableiten der Software Architektur
Camera + Radar ECU

Auf Basis der Requirements der Systemarchitektur Architektur werden die Eigenschaften der Software Architektur festgelegt.

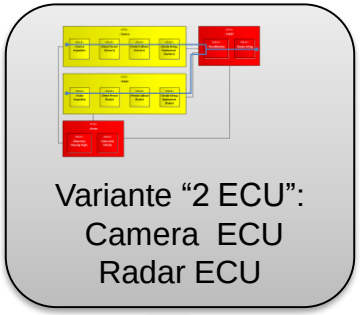
System Architektur

Software Architektur

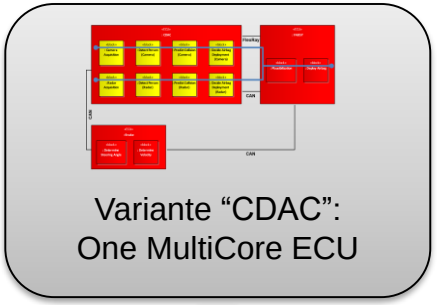


Bewertung der beiden Varianten

Zur **Bewertung** der Varianten wird die Einhaltung der **Requirements** in der **Simulation** mit dem Echtzeitsimulator **chronSIM** untersucht.



Variante “2 ECU”:
Camera ECU
Radar ECU



Variante “CDAC”:
One MultiCore ECU

CHRONSIM

?

Requirements Systemarchitektur

Order	Name
1	RTOS Failures rtos_fail
2	Zeitanforderung $\Delta t \leq 150$ ms
3	Latency(i->j) from "part1 in mainController() on AirbagECU" to "part2 in mainController() on AirbagECU" < 75 ms
4	Event Chain "detFire" Break Off at step max (ignoring last 1)

CHRONSIM

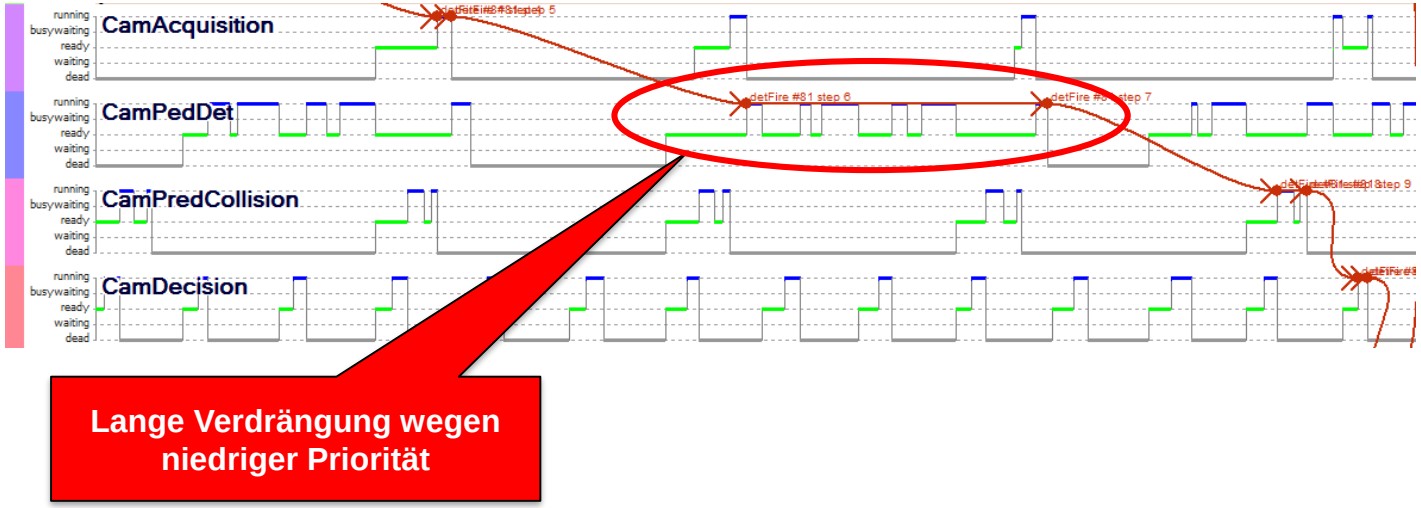
?

Bewertung der Variante Camera + Radar ECU

Es treten viele Requirementverletzungen auf:

Order	Name	State	Failed	Successful	Critical
1	RTOS Failures rtos_fail		0	-	-
2	Event Chain "detFire" Latency step 0 to 18 <= 150 ms		1.3% (1)	98.7% (75)	3.9% (3)
3	Latency(i->j) from "part1 in plausibilization() on FabsyEcu" to "part2 in plausibili...		17.4% (16)	82.6% (76)	8.7% (8)
4	Event Chain "detFire" Break Off at step max (ignoring last 1)		17.6% (16)	82.4% (75)	-

Tasks-Zustandsdiagramm für Detailanalyse

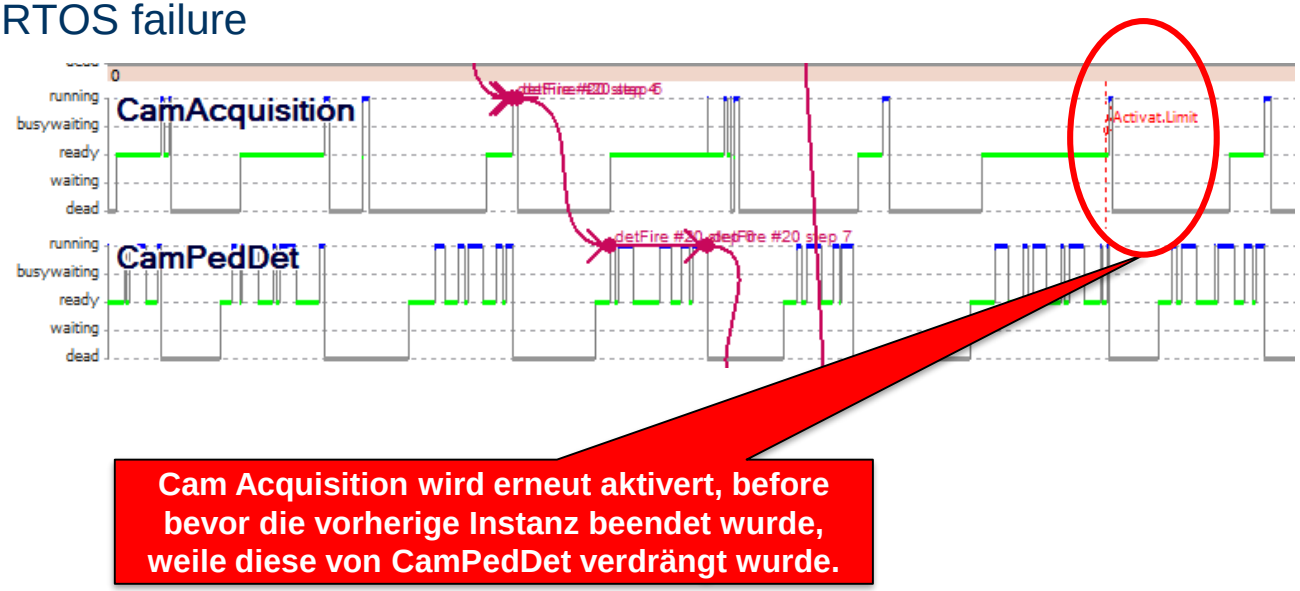


Optimization: Priority

Idee: Erhöhung der Priorität von “Pedestrian Detection” (10➔25), um die Dauer der Verdrängungen zu reduzieren.

Order	Name	State	Failed	Successful	Critical
1	RTOS Failures rtos_fail		33	-	-
2	Event Chain "detFire" Latency step 0 to 18 <= 150 ms		0% (0)	100% (110)	7.3% (8)
3	Latency(i->j) from "part1 in plausibilization() on FabsyEcu" to "part2 in plausibili...		12.7% (16)	87.3% (110)	11.9% (15)
4	Event Chain "detFire" Break Off at step max (ignoring last 1)		12.7% (16)	87.3% (110)	-

Es treten weiterhin Requirementverletzungen auf, jetzt auch mit zusätzlichen RTOS failure



Cam Acquisition wird erneut aktivert, before bevor die vorherige Instanz beendet wurde, weile diese von CamPedDet verdrängt wurde.

Wesentliche Freiheitsgrade Software-Architektur

Festlegung Anzahl Tasks und Interrupts

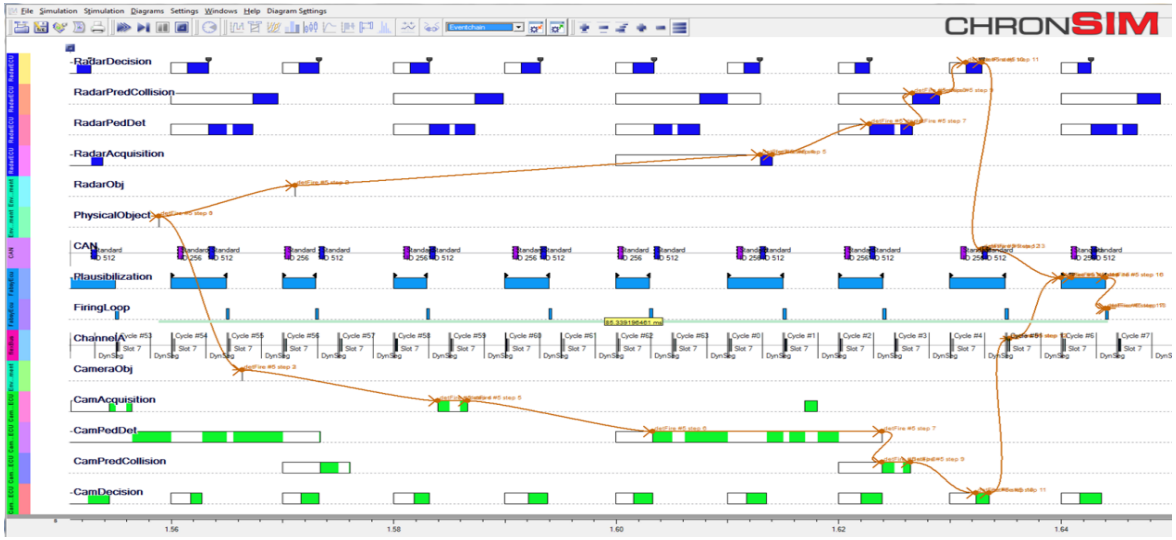
Mapping von Software-Funktionen auf Tasks und Interrupts

Mapping von Tasks und Interrupts auf Cores

Definition von Priorität und Periode für Tasks und Interrupts

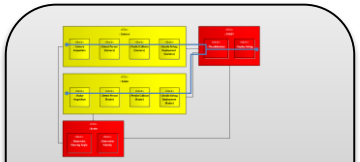
Festlegung Schedulingeigenschaften der Tasks und Interrupts
(Prioritäten, Start-Offset, Unterbrechbarkeit)

Wirkketten: Ablaufreihenfolge der Wirkkettenschritte

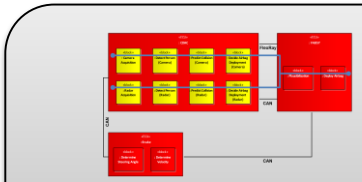


Optimierung der beiden Varianten

Zur **Optimierung** und **Bewertung** der Varianten wird die Einhaltung der **Requirements** in der Simulation mit dem Echtzeitsimulator **chronSIM** untersucht.



Variante “2 ECU”:
Camera ECU
Radar ECU



Variante “CDAC”:
One MultiCore ECU

CHRONSIM

CHRONSIM

Requirements Systemarchitektur

Order	Name
1	RTOS Failures rtos_fail
2	Zeitanforderung Δt ≤ 150 ms
3	Latency(i->j) from "part1 in mainController() on AirbagECU" to "part2 in mainController() on AirbagECU" < 75 ms
4	Event Chain "detFire" Break Off at step max (ignoring last 1)

Name
RTOS Failures rtos_fail
Event Chain "detFire" Latency step 0 to 18 <= 150 ms
Latency(i->j) from "part1 in plausibilization() on FabsyEcu" to "part2 in plausibili...
Event Chain "detFire" Break Off at step max (ignoring last 1)

Name
RTOS Failures rtos_fail
Event Chain "detFire" Latency step 0 to 18 <= 150 ms
Latency(i->j) from "part1 in plausibilization() on FabsyEcu" to "part2 in plausibili...
Event Chain "detFire" Break Off at step max (ignoring last 1)

Optimierung der beiden Varianten

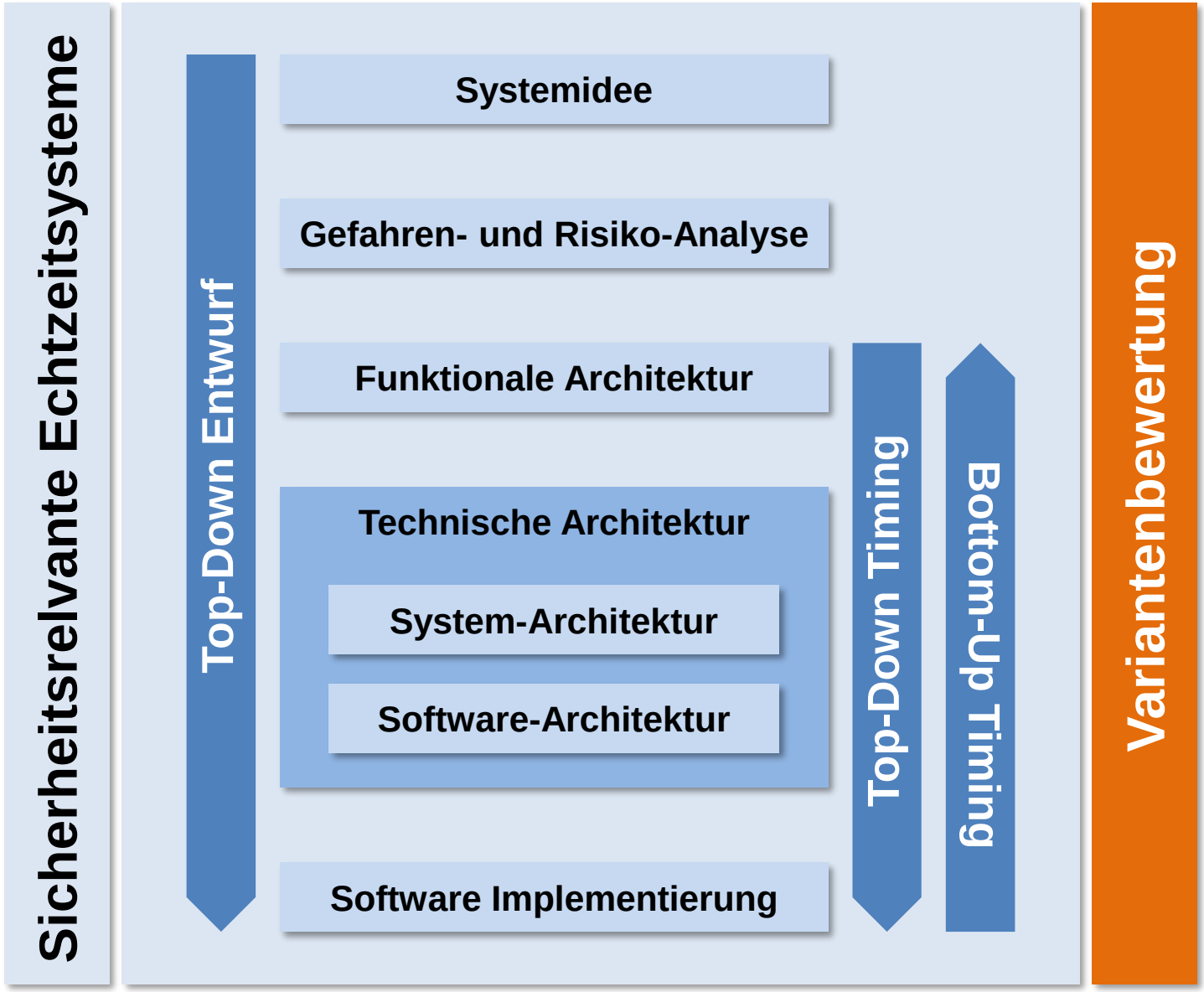
Zur **Optimierung** und **Bewertung** der Varianten wird die Einhaltung der **Requirements** in der Simulation mit dem Echtzeitsimulator **chronSIM** untersucht.

Beide Varianten können die Echtzeitanforderungen erfüllen

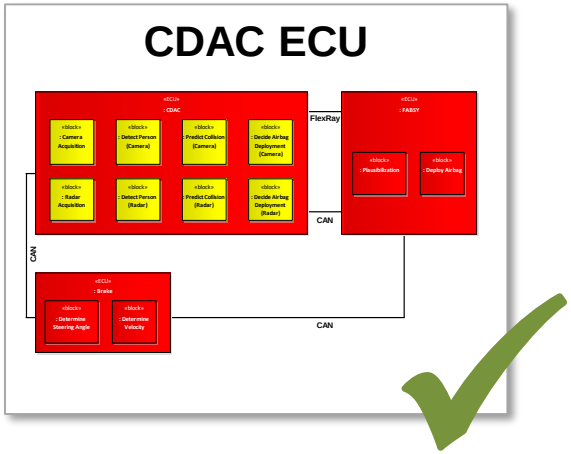
Die Variante "CDAC ECU" (Multicore) kann Synergieeffekte nutzen, da nur eine BSW benötigt wird

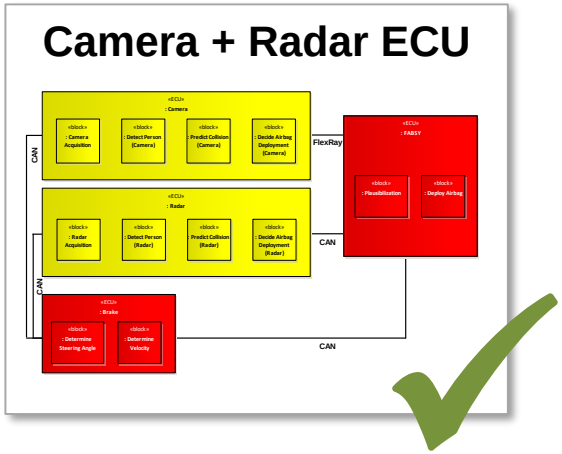
Die Variante "CDAC ECU" ermöglicht eine optimierte Verteilung der Tasks, die bei der Variante "Camera + Radar ECU" nicht möglich ist.

Überblick


Sicherheitsrelvante Echtzeitsysteme
Top-Down Entwurf
Systemidee
Gefahren- und Risiko-Analyse
Funktionale Architektur
Technische Architektur
System-Architektur
Software-Architektur
Software Implementierung
Top-Down Timing
Bottom-Up Timing
Variantenbewertung

Variantenbewertung





Werden Echtzeit- und Safety-Anforderungen erfüllt?

Auswahl zwischen verbleibenden Varianten anhand weiterer Kriterien

Gesamtkosten Stückliste

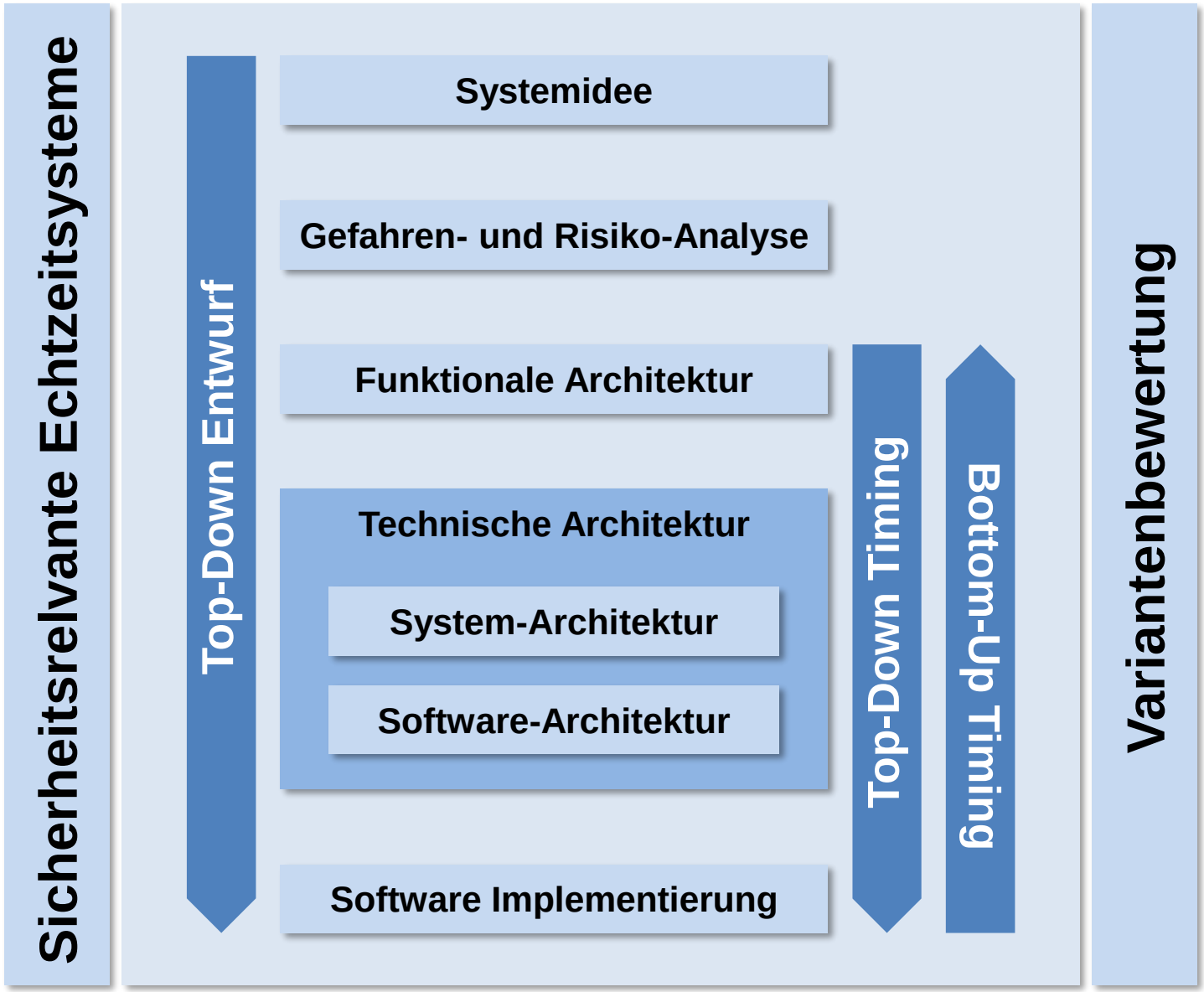
Bauraum

Entwicklungsaufwand

Gewicht

Lieferantenbeziehungen

Zusammenfassung

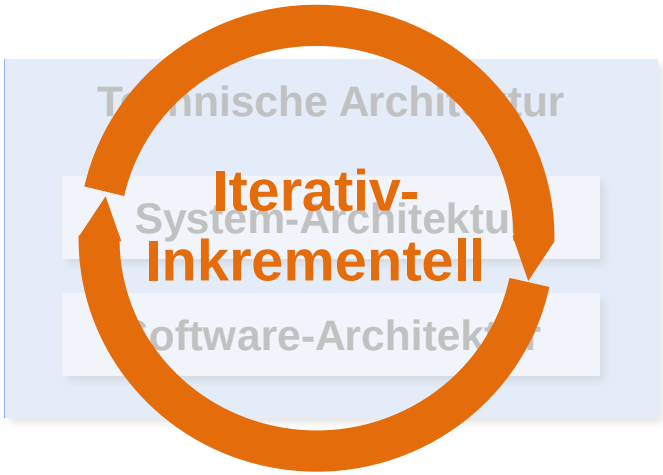


Zusammenfassung

Funktionale Architektur → vermeidet vorschnelle technische Festlegung

Top-Down

Modellbasierter Entwurf mit Timing-Analysen ermöglicht frühzeitige Architekturbewertung



Verfeinerung der Modelle durch Bottom-Up Timing-Informationen

Bottom-Up

- Frühzeitige Gewissheit über Erfüllung von Safety- und Echtzeit-Anforderungen
- Kostengünstiger Vergleich von Architekturvarianten
- Wettbewerbsvorteile durch Auswahl der optimaler Architekturvariante

Vielen Dank!

Ulrich Becker
Method Park

Isabella Stilkerich
Schaeffler Technologies

Ralf Münzenberger
INCHRON